Escuela Técnica Superior de Ingeniería

Grado en **Ingeniería de Organización Industrial**

Trabajo Fin de Grado

Integración de Raspberry Pi y microcontrolador Pycom para la realización de tareas complejas en un entorno IoT

María Andrades Cózar

Tutor: Pablo Millán Gata y Luis Orihuela Espina

Sevilla, Julio de 2019

A mis padres, por haberme dado la oportunidad de estudiar en esta universidad, a mi familia, amigos y todas las personas que me han apoyado en esta etapa.

Resumen

Este proyecto consiste en la integración de una Raspberry Pi 3 y un microcontrolador Pycom con la finalidad de realizar una tarea periódica con el menor consumo posible. El estudio se ha enfocado en la industria agrónoma, donde la tarea a realizar sería tomar una foto y realizar ciertos cálculos para tomar decisiones sobre el riego a aplicar. La necesidad de integrar ambos elementos radica principalmente en el consumo energético ya que, en este tipo de aplicaciones remotas, no es posible conectarse a la red eléctrica.

La Raspberry Pi tiene una gran capacidad de procesamiento que le permite realizar tareas complejas, sin embargo, esto supone también un alto consumo de energía. El microcontrolador Pycom, por el contrario, es un microprocesador diseñado especialmente para este tipo de aplicaciones IoT, de forma que su bajo consumo le permite operar durante mucho tiempo desconectado de la red eléctrica, pero su capacidad de procesamiento es limitada. Además, permite emplear la tecnología de comunicación LoRaWAN, con un alcance mucho mayor que aquellas que incorpora la Raspberry Pi.

En este documento se expone cómo se ha realizado tanto la conexión de la Raspberry Pi con el microcontrolador, como la de este último con The Things Network, un servidor red al que es posible enviar datos a través de la tecnología LoRaWAN. Se presentan posibles aplicaciones y líneas de investigación a seguir en un futuro para mejorar la aplicación inicial.

Palabras clave: Raspberry Pi, Pycom, LoRaWAN, The Things Network, Internet de las Cosas.

Abstract

This project is about the integration of a Raspberry Pi 3 and a Pycom microcontroller in order to perform a periodic task with the lowest possible consumption. The study has been focused on the agronomic industry, where the task to be performed would be to take a photo and perform certain calculations to make decisions about the irrigation to apply. The need to integrate both elements lies mainly in the energy consumption since, in this type of remote applications, it is not possible to be connected to the electricity grid.

The Raspberry Pi has a large processing capacity that allows it to perform complex tasks, however, this also means a high-power consumption. The Pycom microcontroller, on the other hand, is a microprocessor designed especially for this type of IoT applications, so that its low consumption allows it to operate for a long time disconnected from the grid, but its processing capacity is limited.
In addition, it allows the use of LoRaWAN communication technology, with a much greater range than those incorporated in the Raspberry Pi.

This document shows how the connection of the Raspberry Pi with the microcontroller has been made, as well as the connection of the Pycom with The Things Network, a network server to which it is possible to send data through LoRaWAN technology. Possible applications and lines of research to be followed in order to improve the initial application are presented.

Keywords: Raspberry Pi, Pycom, LoRaWAN, The Things Network, Internet of Things.

Índice

Índice de tablas

Índice de figuras

1. Introducción

El IoT o “Internet de las Cosas” consiste en una red de dispositivos interconectados entre sí y con internet. Es decir, estas “cosas” -ya sean objetos de la vida cotidiana, vehículos, animales en una granja o cultivos en un invernadero- tendrían la posibilidad de recopilar, enviar o recibir datos, convirtiéndose en dispositivos inteligentes.

Según IERC (Internet of Things European Research Cluster), “el Internet de las Cosas es una infraestructura de red global dinámica con capacidades de autoconfiguración basadas en protocolos de comunicación estándar e interoperables donde las cosas físicas y virtuales tienen identidades, atributos físicos y personalidades virtuales; utilizan interfaces inteligentes y se integran a la perfección en las redes de información.” (Vermesan & Friess, 2014)

Desde hace varios años es muy común el uso de esta tecnología en ciertas aplicaciones como, por ejemplo, en la domótica. Sin embargo, a raíz de la llamada Cuarta Revolución Industrial o Industria 4.0, cada día más industrias en todos los sectores están digitalizándose y aprovechando los beneficios de esta tecnología. Esta revolución implica una nueva forma de producir y operar, y está marcada por la aparición de nuevas tecnologías inteligentes: robótica, nanotecnología, inteligencia artificial, Internet of Things… y pronto también será clave el despliegue del nuevo estándar de comunicación inalámbrica: 5G (Deloitte Insight, 2017).

Se espera que el 5G suponga un gran impulso al Internet de las Cosas, ya que permitirá potenciar las conexiones entre objetos y aportará grandes mejoras y avances en cuanto a funcionalidad y eficacia se refiere (Universidad de Alcalá, 2019). La quinta generación de conexiones inalámbricas promete una arquitectura altamente flexible diseñada para adaptarse a casi cualquier caso de uso en el espacio de IoT donde, en un futuro cada vez más cercano, miles de millones de dispositivos estarán conectados de manera permanente (Grupo Garatu IT Solutions, 2019).

La industria agraria se ha unido a esta revolución industrial, y se está investigando acerca de las muchas consecuencias positivas de la implantación de estas tecnologías. Además, este sector tiene el reto añadido de la necesidad de optimizar recursos, principalmente el agua, pero también fertilizantes, pesticidas… Automatizando ciertos procesos que tradicionalmente se han realizado basándose en la experiencia en lugar de datos, es posible reducir costes, aumentar la producción y la calidad de los productos, contribuyendo a la sostenibilidad medioambiental.

Según las estimaciones sobre la población mundial de las Naciones Unidas, se prevé un crecimiento de 9.8 miles de millones de personas para 2050 (Naciones Unidas, 2017). Este crecimiento supondrá un crecimiento en la demanda de alimento, pero aumentar a esta escala la producción puede ser difícilmente viable en materia de tierra, uso de agua y rendimiento. Por tanto, el desafío es grande: adoptar métodos de producción más eficientes, sostenibles y adaptados al cambio climático. Se espera que la investigación y desarrollo de estas tecnologías en el sector pueda ayudar a conseguir estos objetivos (Organización de las Naciones Unidas para la Alimentación y la Agricultura, 2018).

1.1. Estructura del documento

En este apartado se describe brevemente el contenido de cada una de las secciones de este documento, con objeto de facilitar su lectura y comprensión.

- Capítulo 1: Introducción.

Se explica el problema que existe en la actualidad respecto a este tema, la motivación de este proyecto y los objetivos planteados. También se presenta el desglose de las distintas tareas en las que se dividió inicialmente el proyecto y su planificación en el tiempo, un resumen del material específico necesario y el presupuesto global del proyecto.

- Capítulo 2: Estado del arte.

Revisión de la literatura existente y de aplicaciones similares utilizadas en la actualidad.

- Capítulo 3: Herramientas utilizadas.

En este capítulo se describen en líneas generales las herramientas tanto de hardware como de software que fueron utilizadas para el desarrollo del proyecto.

- Capítulo 4: Desarrollo del proyecto.

Esta sección está dividida en los distintos pasos llevados a cabo durante la realización del trabajo. Estos pasos siguen una estructura similar: posibles soluciones, qué solución se escogió y por qué, y cómo se llevó a la práctica. Se mencionan imprevistos y dificultades encontradas.

- Capítulo 5: Resultados.

Se comentan y analizan los resultados obtenidos tras la realización de la aplicación.

- Capítulo 6: Posibles aplicaciones.

En este capítulo se detalla cómo se realizaría la implantación de esta tecnología en un caso real, las instalaciones de Surexport. Se analizan las consecuencias positivas y negativas que tendría dicha implantación, y otras aplicaciones en las que podría ser de utilidad esta investigación.

- Capítulo 7: Conclusiones.

Finalmente se exponen brevemente unas conclusiones personales sobre la realización del proyecto, así como revisión de la consecución de los objetivos marcados inicialmente, así como las investigaciones y desarrollos que se podrían realizar a continuación de este proyecto.

1.2. Motivación

La idea de este trabajo de fin de grado surge a raíz del planteamiento de un proyecto en el cual participa la Universidad Loyola Andalucía: “INNORIEGA” (rIego sosteNible basado en moNitORización distribuida e intEliGencia Artificial), PP.AVA.AVA.2019.024, I.P. Pedro Gavilán Zafra, 2019-2021.

El proyecto tiene como objetivo generar un modelo digital para gestionar de forma inteligente y sostenible el riego necesario para los cultivos utilizando nuevas tecnologías. Con ello se pretende que el sector agrícola en Andalucía se sume a la llamada “Agricultura Inteligente” y que la producción sea más eficiente, segura y sostenible.

Se llevará a cabo en las instalaciones de Surexport S.L. (empresa productora y exportadora de *berries*) y contará con un equipo formado por investigadores de IFAPA, CSIC y Universidad Loyola Andalucía, además de la colaboración de empresas como EasytoSee Agtech.

Realizar un Trabajo de Fin de Grado en esta línea resultó ser muy atractivo para la alumna, no solo por la oportunidad de profundizar en el estudio de este tipo de tecnologías que están en la vanguardia de la industria, sino también por la importancia medio ambiental de las investigaciones en este ámbito.

1.3. Objetivos

Un primer objetivo general del proyecto es claro: integrar una Raspberry Pi 3 Modelo B y un microcontrolador Pycom LoPy4 de forma que puedan realizar tareas complejas con el menor consumo energético posible y desconectado de la red eléctrica y de Internet. De ello se deriva un segundo objetivo, que es el envío de información a equipos remotos sin utilizar tecnologías que requieran de un coste periódico, como 4G, o un equipo con alimentación permanente.

De estos objetivos generales se pueden desglosar algunos objetivos específicos:

- Encontrar la mejor forma de transmisión del dato calculado por la Raspberry Pi al microcontrolador Pycom. Esta solución debe ser robusta e independiente de la red Wi-Fi.
- Profundizar en la tecnología LoRa, una tecnología emergente e idónea para Internet of Things.
- Optimizar el tiempo de trabajo del conjunto, de forma que el sistema pase el mayor tiempo posible en reposo; pero sin dejar de lado la necesidad de obtener una información fiable.
- Servir como base o apoyo para otros proyectos de investigación en este ámbito.

1.4. Planificación

1.4.1. Desglose de tareas

El proyecto tiene como fecha de inicio el 1 de febrero de 2019, y las tareas y subtareas en las que se dividió para su realización son las siguientes:

Tarea 1: Revisión bibliográfica, comprensión finalidad del proyecto y planificación.

Tarea 2: Raspberry Pi.

- Tarea 2.1: Instalar sistema operativo a Raspberry Pi 3 y aprender a usar sus funciones básicas.
- Tarea 2.2: Tomar foto con la cámara de la Raspberry Pi 3 y guardarla.
- Tarea 2.3: Ejecución automática de un *script* con el arranque del sistema operativo.
- Tarea 2.4: Programa con Python + OpenCV de visión artificial que cuente el número de pixeles verdes de la foto tomada por la Raspberry.

Tarea 3: Pycom.

- Tarea 3.1: Leer documentación relevante Pycom
- Tarea 3.2: Configurar tarjeta y probar un programa sencillo

Tarea 4: Transmisión de datos Raspberry Pi 3-LoPy4.

- Tarea 4.1: Elección del método: Bluetooth, serial, MQTT…
- Tarea 4.2: Documentación sobre funcionamiento del método de comunicación escogido (comunicación serial).
- Tarea 4.3: Establecer comunicación serial para enviar mensaje sencillo.
- Tarea 4.4: Establecer comunicación serial de forma que al encender Raspberry, ésta ejecute el programa (a su inicio) y envíe el dato del resultado (es decir, número de pixeles verdes de la foto tomada).

Tarea 5: Enviar datos con LoRaWAN a The Things Network (TTN).

- Tarea 5.1: Información acerca de esta tecnología y su uso.
- Tarea 5.2: Configuración Gateway Sentrius.
- Tarea 5.3: Creación de aplicación para recibir datos de la tarjeta LoPy.
- Tarea 5.4: Programar tarjeta para que envíe los datos recibidos de la Raspberry.
- Tarea 5.5: Prueba del mecanismo completo hasta este punto (transmisión de datos unidireccional RPi-Pycom-TTN).
- Tarea 5.6: Decodificación de datos en The Things Network y forma de archivar los mismos.

Tarea 6: Encendido/apagado Raspberry Pi.

- Tarea 6.1: Montaje circuito con relé para apagar/encender la RPi por órdenes de Pycom.
- Tarea 6.2: Programación de Pycom para que envíe 1/0 por un pin de salida.
- Tarea 6.3: Programar y probar ciclo completo (Pycom enciende RPi, que ejecuta programa y envía dato a Pycom, una vez recibido este apaga la RPi y envía dato a TTN. Finalmente, el microcontrolador Pycom entra en modo reposo hasta nueva repetición del ciclo).
- Tarea 6.4: Montaje de circuito eléctrico en placa de PCB.

Tarea 7: Pruebas con el sistema.

Tarea 8: Elaboración memoria final.

1.4.2. Diagrama de Gantt

En la Figura 1 se muestra en detalle la organización de las distintas tareas durante el desarrollo del proyecto. Para ello se emplea un diagrama de Gantt con el tiempo real empleado para la realización de las tareas definidas con anterioridad.

Los colores van en función del grado de dificultad encontrado a la hora de realizar la tarea o subtarea, siendo el verde un grado de dificultad bajo, amarillo dificultad media y rojo un grado de dificultad elevado.

1.5. Presupuesto

Al realizar el presupuesto se han tenido en cuenta costes de hardware y software, así como el coste de investigación, programación y montaje.

Remarcar que el presupuesto se ha calculado para un sistema Raspberry Pi-Pycom que envía datos a la nube a través de un *gateway*. En el caso de querer montar una red de dispositivos que envíen datos, por ejemplo, uno en cada invernadero, solo sería necesaria la combinación Pycom-Raspberry Pi, pues el *gateway* a través del cual enviarían los datos sería el mismo. Es decir, el equipamiento que habría que multiplicar por el número de nodos que se necesiten es el de la sección “Hardware por nodo”, los otros componentes solo será necesario adquirirlos una vez.

El presupuesto total asciende a una cantidad de quinientos siete con cuarenta y nueve euros (507,49 €), siendo el coste de cada nodo de ciento setenta y cuatro euros con sesenta y un céntimos (174,61 €). El resto hasta el presupuesto total corresponde al coste del hardware de adquisición única.

Se adjunta el presupuesto completo en el Anexo I.

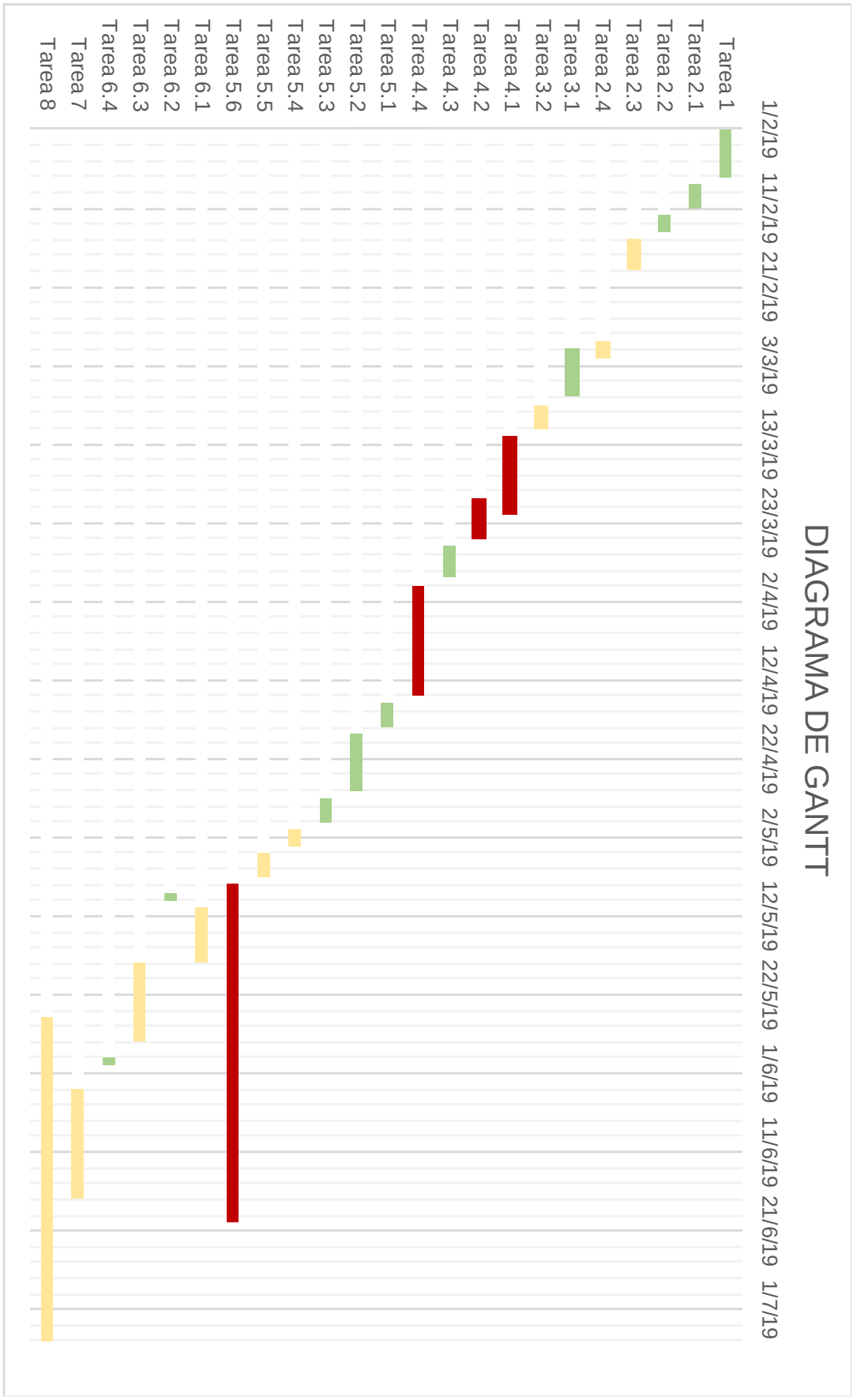


Figura 1: Diagrama de Gantt

Fuente: Elaboración Propia

2. Estado del arte

En este capítulo se hará una descripción en líneas generales del tipo de aplicaciones similares a la que se pretende crear usadas en la actualidad en IoT. Se estructurará en dos partes: hardware y comunicaciones.

2.1. Hardware

El uso de la Raspberry Pi (Raspberry Pi, 2019) para aplicaciones en Internet of Things está muy extendido debido a la alta capacidad de procesamiento, bajo coste y reducido tamaño de ésta. Estas características la hacen muy adecuada para aplicaciones de este tipo, y existe una gran cantidad de bibliografía sobre la configuración de varias Raspberry Pi para crear redes domésticas de bajo coste o redes a pequeña escala (Venkat Subramanian, Krishnamoorthy, & Suleman Khalid, 2017). También está muy generalizado el uso de Arduino (Arduino, 2019) por los mismos motivos, o sistemas compuestos por Arduino y Raspberry Pi (Sheikh & Xinrong, 2014).

De cualquier forma, a la hora de desarrollar un proyecto IoT es necesario tener claros los requisitos que se necesitan, para así poder elegir las plataformas de hardware que cumplan estas características. Por tanto, se realizará una revisión de los dispositivos de hardware comerciales disponibles, enumerando sus características técnicas principales.

A su vez, los dispositivos de hardware se van a dividir en dos grupos: microcontroladores y computadoras de placa única. Las especificaciones técnicas de ambos grupos se van a realizar de la misma forma, es decir, teniendo en cuenta las características clave comunes a la mayoría de estos dispositivos, de forma que se puedan comparar y analizar ventajas y desventajas. Estas características son (IBM Developer, 2018):

- **Adquisición y control de datos.** Incluye los procesos relacionados con la lectura de las medidas físicas proporcionadas por los sensores, y su transformación a señales digitales para su posterior análisis y procesamiento.

- **Procesamiento y almacenamiento de datos.** Los dispositivos IoT pueden, o bien realizar un análisis de los datos, o bien transmitirlos directamente a otros dispositivos como *routers*. La ventaja de procesar y analizar los datos en redes con gran cantidad de dispositivos conectados radica en la posibilidad de enviar en sentido ascendente solo los datos más destacados, evitando el colapso de la red. Los dispositivos que realicen análisis de datos más complejos requerirán una capacidad de procesamiento mayor que aquellos que solo realicen operaciones básicas de análisis.

- **Conectividad.** Puesto que el internet de las cosas se basa en la interconexión de dispositivos, esta es una de las características más importantes a tener en cuenta. Se contemplan tanto comunicaciones inalámbricas como por cable.

- **Gestión de energía.** De particular interés sobre todo para aplicaciones como la que aquí se trata, en las que se usan dispositivos portátiles que van a necesitar depender de baterías.

Como ya se ha comentado, en primer lugar, se enumerarán las especificaciones técnicas de diferentes microcontroladores que existen actualmente en el mercado. Los microcontroladores son circuitos integrados programables, que pueden contener uno o varios núcleos de procesador, memoria RAM y memoria ROM. Esta última es una memoria de solo lectura programable y borrable, y sirve para almacenar los programas personalizados que ejecutará el microcontrolador. Los más importantes actualmente en IoT, y por ello, los escogidos para realizar la comparación son: Arduino Uno, Espressif Systems ESP8266EX (Espressif Systems, 2019) y LoPy 4 (Pycom, 2019).

Característica	Prestación	Arduino Uno	ESP8266EX	LoPy 4
Adquisición y control de datos				
	Pines GPIO	14 e/s digitales (de los cuales 6 son salidas PWM) y 6 entradas analógicas	16 digitales (pueden configurarse como PWM) 1 analógico	Hasta 24 pines GPIO y 8 analógicos
	Voltaje operativo	5 V	2.5-3.6 V	3.3 V-5.5 V
Procesamiento y almacenamiento de datos				
	Procesador	ATMega328PU	32 bit Tensilica L106	ESP32 Dual Core Microcontroller and WiFi/Bluetooth 4.2 radio
	Memoria	32 kB falsh, 1 kB EPROM	64 kB de RAM para instrucciones y 96 kB de datos	4 MB RAM, 8 MB flash
Conectividad				
	Interfaces de red	Ninguna por defecto (se pueden añadir módulos)	WiFi integrado	WiFi, LoRa, SigFox, Bluetooth
Gestión de energía				
	Consumo medio	50 mA	WiFi: 100 mA; 10 mA Deepsleep	Idle: 34,5 mA; Deepsleep: 18,5 uA; Wifi: 104 mA; Lora: 108 mA
Otros				
	Dimensiones	68.6x53.4mm	14.3x24.8 mm	55x20 mm
	Precio aproximado	20 €	3 €	34.95 €

Tabla 1: Comparación microcontroladores

Fuente: Elaboración Propia

Por otro lado, se encuentran los SBC (*Single Board Computers* o Computadoras de Placa Única). Éstas tienen por lo general mayor memoria y capacidad de procesamiento que los microcontroladores, y también la posibilidad de conectar teclados, ratón, monitor… A continuación, se enumeran en la Tabla 2 las características más importantes de tres SBC: Raspberry Pi 3 Modelo B, BeagleBone Black (BeagleBoard.org) y Qualcomm DragonBoard 410c (Developer Qualcomm, 2019).

Característica	Prestación	Raspberry Pi 3 Model B	BeagleBone Black	Qualcomm DragonBoard 410c
Adquisición y control de datos				
	Pines GPIO	40	65	12
	Voltaje operativo	5 V	5 V	1.8 V
Procesamiento y almacenamiento de datos				
	Procesador	ARM Cortex A53	AM335X ARM Cortex A8	ARM Cortex A53
	Memoria	1Gb	4 Gb	1 Gb, 8 Gb flash
Conectividad				
	Interfaces de red	Wifi, Ethernet, Bluetooth	Ethernet, y puertos externos que permiten WiFi y Bluetooth externos	WiFi, Bluetooth, GPS
Gestión de energía				
	Alimentación recomendada	5V 2.5A	5V 1.2A-2A	6.5-18V 2A
Otros				
	Dimensiones	85x56 mm	89x54.5 mm	55x20 mm
	Precio aproximado	33.95 €	56.65 €	75 €

Tabla 2: Comparación SBC

Fuente: Elaboración Propia

2.2. Comunicaciones

En cuando a protocolos de transmisión de datos en IoT se ha venido usando de forma generalizada Zigbee (ZigBee Alliance, 2019), sin embargo, desde hace poco ha ido tomando más fuerza la tecnología LoRa (LoRaWAN España, 2019); esta utiliza un rango de frecuencias más bajo y es capaz de cubrir grandes distancias. ZigBee es especificación de un conjunto de protocolos de alto nivel de comunicación basada en el estándar IEEE 802.15.4.

LoRa pertenece al grupo de tecnologías llamadas LPWAN (*Low Power Wide Area Network* o Red de Área Amplia a Baja Frecuencia). LPWAN es un protocolo de conexiones inalámbricas caracterizado por tener bajo coste, bajo consumo y amplio rango de operación. Además de LoRaWAN, existen otras tecnologías importantes dentro de las LPWAN: SigFox (SigFox España, 2019) y NB-IoT (Accent Systems NB IoT, 2019).

Estas características de las tecnologías LPWAN marcan la diferencia con otras como ZigBee, haciéndolas más adecuadas para su uso en Internet of Things. ZigBee tiene una topología de malla, esta clase de redes son útiles en distancias medias y muchos sistemas de automatización del hogar se implementan usando dicha tecnología, sin embargo, no tienen las capacidades de largo alcance que pueden proporcionar las redes LPWAN. Además, en estas redes cada nodo debe recibir y repetir constantemente las señales vecinas, por lo que cuando los sensores o nodos se escalan a miles, este tipo de redes no se ajustan de forma adecuada ni son eficientes en términos de energía (McCelland, 2017).

Se realizará por tanto a continuación -ver Tabla 3- una comparación de las otras tres importantes tecnologías en el panorama del IoT: LoRa, SigFox y NB-IoT (Link Labs, 2018), (Barry & Meijers, 2018).

Se puede concluir que cada tecnología será más adecuada dependiendo de la aplicación IoT que se pretenda realizar. SigFox y LoRa tienen precios más bajos, un alcance muy grande y una vida útil muy larga. Además, LoRa a diferencia de SigFox permite una comunicación más fiable con dispositivos en movimiento. Por el contrario, NB-IoT será de mayor utilidad en mercados de mayor valor que estén dispuestos a

pagar un precio más alto a cambio de una alta calidad del servicio (Mekki, Bajic, Chaxel, & Meyer, 2019).

Característica	LoRa	SigFox	NB-IoT
Espectro de frecunecias	Banda ISM sin licencia(868 MHz en Europa)	Banda ISM sin licencia(868 MHz en Europa)	Necesita un canal específico dedicado, con licencia
Ancho de banda	125-250 kHz	100 Hz	200 kHz
Velocidad de transmisión de datos	50 kbps	100 bps	200 kbps
Cobertura	5 km medio urbano, 20 km rural	10 km medio urbano, 40 km rural	1 km medio urbano, 10 km rural
Inmunidad ante interferencias	Muy alta	Muy alta	Baja
Seguridad	Basado en sesiones, en cada inicio de sesión se intercambian una serie de claves para dicha sesión	Por defecto, los datos se transmiten en el aire sin ningún cifrado; se puede cifrar de extremo a extremo en la capa de aplicación	Se usan múltiples identificadores, claves y métodos de cifrado en los diferenc¡tes niveles del protocolo
Es buena para…	Ejecutar una red aislada o privada en el medio rural o la ciudad. Es ideal para sensores que no necesitan enviar constantemente.	Lectura remota de contadores de electricidad o agua.	Actualización de sistemas basados en GSM. Principalmente para lecturas de sensores, seguimiento y gestión de flotas.
No funciona bien en…	No es adecuada para aplicaciones en las que se requiere una respuesta inmediata	La cantidad de mensajes al día está limitada, por lo que no es adecuada en aplicaciones que requieran un envío de gran cantidad de mensajes	Aplicaciones en las que se necesite enviar un gran volumen de datos

Tabla 3: Comparación tecnologías IoT

Fuente: Elaboración Propia

Tal y como se ha señalado en la tabla, uno de los puntos fuertes de LoRa es su alta inmunidad a las interferencias, de hecho, diversos estudios han demostrado esto, proporcionando además buenos resultados en usos similares a los que se pretenden llevar a cabo en este proyecto (Jedermann, y otros, 2018).

2.3. Justificación de la elección

Lo conveniente en este caso es elegir en un primer lugar el tipo de tecnología o protocolo que se quiere utilizar para establecer las comunicaciones en la red IoT, y posteriormente elegir la combinación de elementos de hardware que permitan dicha comunicación.

Teniendo en cuenta lo expuesto anteriormente, para el proyecto que se quiere realizar se podrían descartar tanto ZigBee como NB-IoT; la primera por el hecho de que se va a necesitar cubrir amplias distancias, y la segunda porque el coste es más elevado tanto de instalación como por nodo, y en el futuro la idea sería poder escalar esta red a un gran número de sensores y nodos situados en diferentes partes de la instalación.

Entre SigFox y LoRa se decidió escoger LoRa, ya que es una tecnología libre en una banda libre, es decir, permite a cualquiera montar su propia red y conectarla a The Things Network (The Things Network Community, 2019). Esto no ocurre con SigFox pues, aunque opera en una banda de frecuencias libre, es una tecnología propietaria (VGD, 2018).

Una vez realizada dicha elección, se decide usar el microcontrolador Pycom LoPy4, ya que tiene integrada la comunicación a través de LoRa, sin necesidad de agregar ningún módulo extra. Dado que este dispositivo no es capaz de realizar las tareas que se requieren, y la imposibilidad de acoplar un módulo de cámara, surge la necesidad de utilizar este microcontrolador de forma conjunta con una SBC. La Raspberry Pi 3 Modelo B es una de las SBC más baratas que cumple con las características requeridas.

3. Herramientas utilizadas

En esta sección se pretende plasmar una descripción en términos generales, sin llegar a ser exhaustiva, de las principales herramientas empleadas para el desarrollo del proyecto.

3.1. Software

Como se verá a continuación, todas las herramientas de software utilizadas son libres, lo cual facilita que cualquier usuario pueda seguir investigando y profundizando en esta línea.

3.1.1. Python/MicroPython

Prácticamente todo el trabajo ha sido programado en lenguaje Python (Python Programming, 2019). Es un lenguaje de programación interpretado, de alto nivel, de código abierto y multiplataforma.

En un primer momento se eligió este lenguaje para la programación de la Raspberry Pi por la sencillez de su sintaxis y por ser de código abierto, es decir, no requiere licencia. También fue un hecho decisivo en la elección de este software la posibilidad de instalar la biblioteca de visión artificial OpenCV. Se ha utilizado la versión Python 2.7 en la Raspberry Pi.

En el caso de la tarjeta Pycom se ha utilizado MicroPython (MicroPython, 2019), un intérprete pequeño pero eficiente de Python 3 y creado específicamente para la programación de microcontroladores.

El editor de código escogido en la programación de Pycom ha sido Atom (Atom, 2019), puesto que también es de código abierto, y tiene la posibilidad de instalar el paquete

pymakr (Atom Packages, 2019). Este paquete permite conectarse al microcontrolador desde la consola de Atom en un ordenador, pudiendo probar el código, ejecutar comandos o cargar proyectos enteros a la tarjeta Pycom.

3.1.2. OpenCV

OpenCV (*Open Source Computer Vision Library*) es una biblioteca de software de visión de computadora y de aprendizaje automático (*Machine Learning)* de código abierto (OpenCV, 2019).

Tiene una gran cantidad de opciones de tratamiento y procesamiento de imágenes y su uso está muy extendido en el campo de la visión artificial. Es también multiplataforma y es compatible con casi todos los lenguajes de programación.

Aunque en este proyecto en concreto no se trata en profundidad la visión artificial, y probablemente no hubiera hecho falta una biblioteca tan específica, la idea es que en un futuro se complemente con otro Trabajo de Fin de Grado que se está llevando a cabo también en la Universidad Loyola Andalucía. Éste es sobre "*Sistemas de visión estéreo para obtener profundidad de objetos",* de forma que en la Raspberry Pi se ejecutarían estos algoritmos en lugar de los que aquí se presentan.

3.1.3. The Things Network (TTN)

Finalmente sería interesante presentar aquí la iniciativa The Things Network, aunque no es una herramienta de software en sí.

The Things Network pretende convertirse en una red global para el IoT, es de uso libre y de su creación, desarrollo y regulación se encargan miles de voluntarios en todo el mundo (LoRa Alliance, 2019).

Usa la tecnología LoRaWAN, permitiendo a las "cosas" conectarse sin necesidad de Internet o Wi-Fi. Gracias a esto los costes son reducidos, al igual que el consumo energético y puede cubrir distancias de hasta decenas de kilómetros.

Este servidor red hace posible ver en directo desde cualquier lugar lo que están transmitiendo los dispositivos o nodos configurados, simplemente accediendo a la página web de The Things Network.

3.2. Hardware

3.2.1. Raspberry Pi 3 Modelo B con módulo de cámara V2 8MP

La Raspberry Pi, ver figura 2, es un SBC (*Single Board Computer*, Ordenador de Placa Única), esto quiere decir, es un ordenador completo a excepción de periféricos: teclado, ratón, monitor… Su tamaño y coste son muy reducidos y se ideó para estimular el aprendizaje de la programación y la informática.

Puede verse un resumen de sus prestaciones en la Tabla 4:

PRESTACIONES RASPBERRY PI 3 MODELO B
Procesador
Chipset Broadcom BCM2387
1,2 GHz 64 bit quad core ARM Cortex-A53
RAM
1GB LPDDR2
Conectividad
Ethernet 10/100BaseT
802.11 b/g/n LAN inalámbrica (WiFi)
Bluetooth 4.1 Low Energy (BLE)
Salida HDMI
Salida de vídeo RCA compuesto
Salida de audio
4 puertos USB 2.0
40 pines GPIO
Conector cámara CSI-2
Ranura micro SD para cargar OS y almacenar datos
Fuente de alimentación micro USB
Alimentación
5 V 2.5 mA

Tabla 4: Prestaciones Raspberry Pi 3 Modelo B

Fuente: Elaboración Propia

Se ha utilizado también el módulo de cámara V2 para tomar las fotos, del mismo fabricante, cuyas características principales se enumeran a continuación:

Figura 2: Raspberry Pi 3 Modelo B + Módulo de cámara V2

- Resolución de 3280x2464 píxeles
- Captura de vídeo a 1080p/30, 720p/60 y 640p x 480p/90
- Sensor de imagen Sony IMX219 de 8MP
- Dimensiones: 25x23x9mm

En la Figura 3 se muestra el mapa de pines GPIO del modelo de Raspberry Pi empleado.

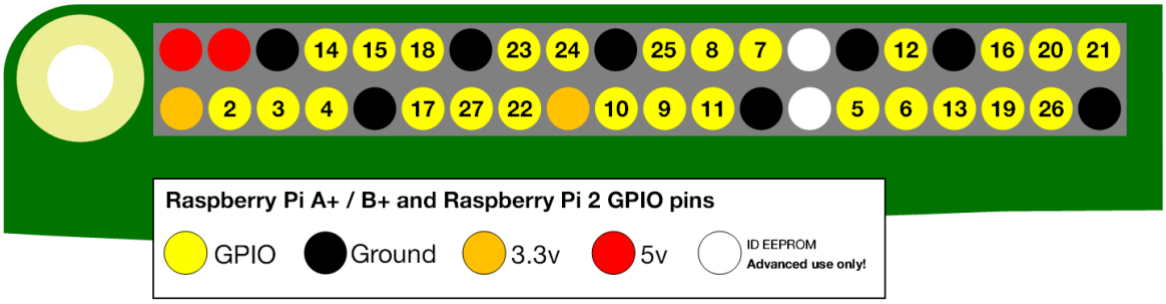


Figura 3:Mapa de pines Raspberry Pi 3 Modelo B

Fuente: Documentation GPIO, 2019

3.2.2. Microcontrolador Pycom

Compuesto por la tarjeta de desarrollo LoPy4 y la placa de expansión 3.0 (ver Figura 4). La tarjeta tiene micropython habilitado y puede transmitir por Bluetooth, WiFi, SigFox y LoRa. La placa de expansión permite la conexión vía USB al ordenador, conexión serial, etcétera. Para una descripción detallada de las prestaciones de la placa de desarrollo LoPy 4, ver Tabla 5.

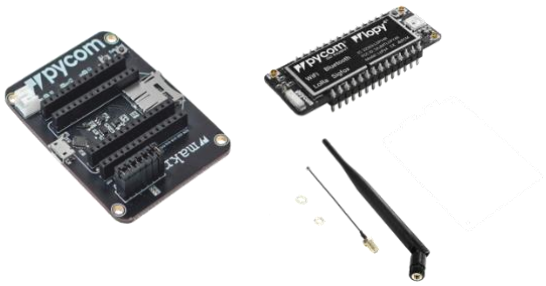


Figura 4: Placas de desarrollo y de expansión Pycom + antena LoRa

Ya que la transmisión de datos se hará a través de LoRa, será necesaria la antena para este fin, pues de lo contrario no serán posibles las transmisiones a largas distancias.

PRESTACIONES LoPy 4
CPU
Microprocesador LX6 de 32 bits de doble núcleo Xtensa Aceleración del punto flotante del hardware Python multi-threading Un coprocesador ULP adicional que puede monitorear GPIOs, los canales ADC y controlan la mayoría de los periféricos internos durante el modo de suspensión profunda, mientras que solo consumen ~ 25uA
Memoria
RAM: 520KB + 4MB Flash externo: 8MB
Comunicaciones
WiFi 802.11b/g/n 16mbps Bluetooth clasico y BLE LoRa SigFox
Dimensiones
55x20x3.5mm
Otros
WS2812 RGB multi–colour LED

Tabla 5: Presataciones LoPy4

Fuente: Elaboración Propia

Se incluye en la Figura siguiente el mapa de pines de la tarjeta Pycom:

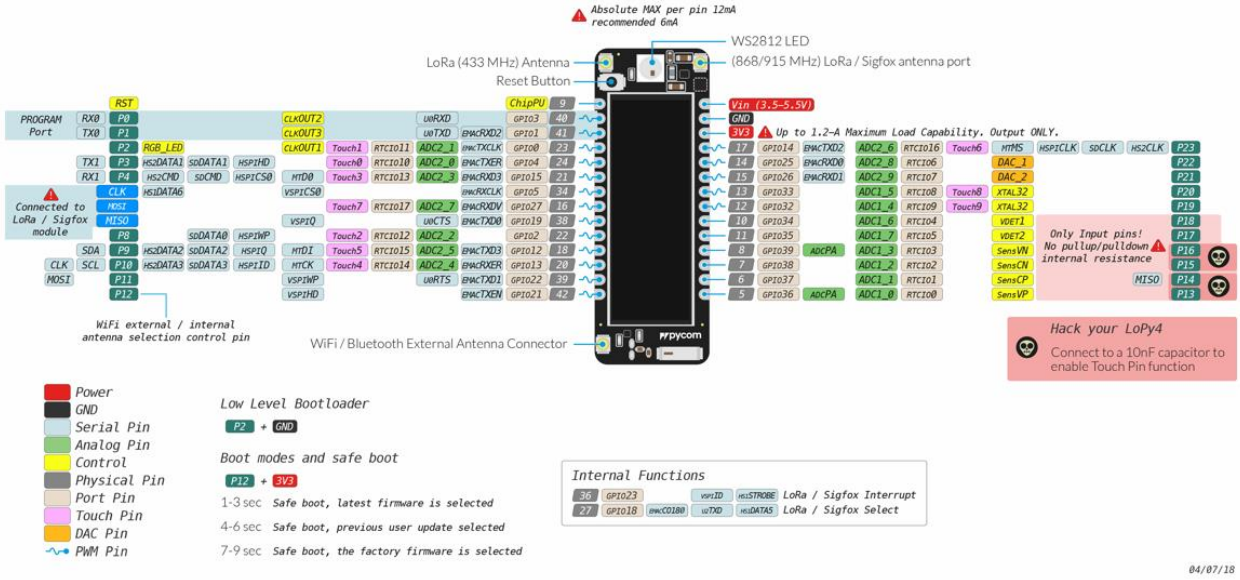


Figura 5: Mapa de pines tarjeta Pycom

Fuente: LoPy4 Datasheet Verson 1.0, 2018

3.2.3. Gateway Sentrius RG1xx

Este *gateway* tiene la función de recibir la información del sistema RaspberryPi-Pycom (nodo) y enviar la información al servidor de red The Things Network. La integración de la Raspberry Pi y el Pycom será un nodo de la red LoRaWAN, cuando se construya la red real dispondremos de muchos nodos -no uno solo- y todos ellos irán enviando información a través de este *gateway*. Posteriormente el *gateway* gestionará los protocolos para enviar la información a un servidor red, en este caso, The Things Network.

Figura 6: Gateway Sentrius RG1xx

Un resumen de sus prestaciones se incluye en la Tabla 6 (Laird Smart Technology, 2019):

PRESTACIONES GATEWAY SENTRIUS SERIES RG1xx
Software
Sistema operativo: Embedded Linux, 4.x Kernel
LoRa
Chipset
LoRa: Semtech SX1301/1257
Bluetooth: Cambridge Silicon Radio CSR8811 A08
WiFi: Qualcomm Atheros QCA6004
Interfaces
Cable: Ethernet - conector RJ45
Inalámbricas: LoRaWAN, WiFi 2.4/5GHz, Bluetooth 4.0 (BLE y clásico)
Protocolos
Semtech Packet Forwarder - Soporte predeterminado para The Things Network, Semtech IoT, Stream Communications, LORIOT, Senet
Alimentación
12 V 1 A
Dimensiones
133x275x30 mm

Tabla 6: Prestaciones gateway Sentrius RG1xx Laird

Fuente: Elaboración Propia

4. Desarrollo del proyecto

Tras haber hecho una descripción de los objetivos, planificación y elementos que componen el sistema, es conveniente en este punto realizar una explicación detallada de cómo se ha llevado a cabo cada uno de los pasos, qué dificultades se han encontrado y cómo se han ido solventando cada uno de estos problemas.

Con el objeto de clarificar el funcionamiento del sistema, se presenta a continuación un esquema de comunicaciones en el que se indican las conexiones que se deben establecer.

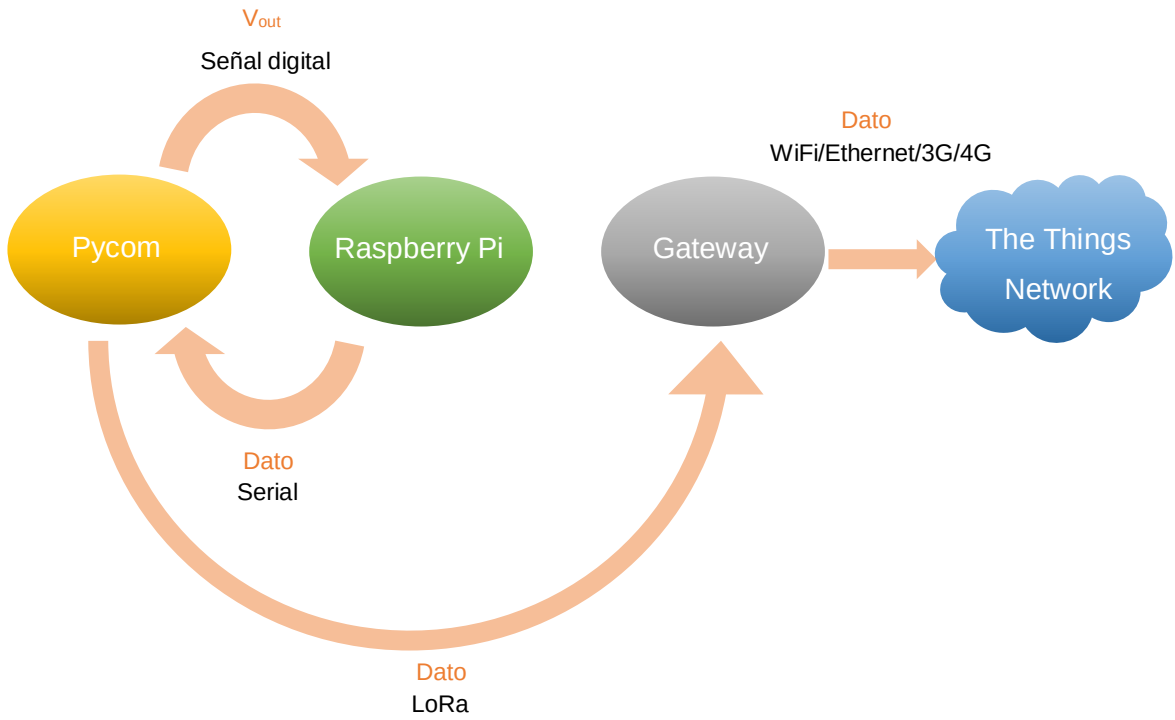


Figura 7: Esquema de conexiones

Fuente: Elaboración Propia

En el esquema de la Figura 7, sobre las flechas aparece en color naranja la información que se envía y en negro el protocolo de comunicación. V_{out} es la señal

digital que envía el microcontrolador a la Raspberry Pi para encenderla o apagarla en función de si esta señal es 0/1. *Dato* es el resultado de la operación que efectúa la Raspberry Pi, y por tanto el dato que interesa enviar a la nube.

4.1. Configuración y programación Raspberry Pi

En este apartado se engloba -siguiendo la planificación del apartado 1.4- la tarea número 2, incluyendo todas sus subtareas.

4.1.1. Instalación del sistema operativo y software necesario

En la configuración inicial e instalación del sistema operativo fueron de mucha utilidad los tutoriales encontrados en la página oficial de Raspberry Pi (Raspberry Pi, 2019).

El sistema operativo escogido fue Raspbian, el oficial de la Fundación Raspberry Pi, que es una distribución de Linux. Habría sido posible la instalación de cualquier otro sistema operativo, como otras distribuciones de Linux o Windows, por ejemplo, pero solo necesitamos que soporte Python y que sea lo más rápido posible. Sobre todo, por este último requisito es preferible el uso de Raspbian, pues es una versión de Debian creada especialmente para este dispositivo, muy estable y completo a la vez que sencillo de usar, y está optimizado para sacar el máximo partido a las funciones de la Raspberry Pi.

Respecto a la cámara, como ya se ha especificado con anterioridad, se usó el módulo V2, de 8 megapíxeles y una resolución de 3280x2464. Este módulo se conecta a al microordenador a través de un cable plano CSI.

Es posible acceder a la cámara desde Python importando la biblioteca "*picamera*". Tras realizar varias pruebas con esta biblioteca, se llega a la conclusión de que es necesario algún módulo de visión por computadora para el procesamiento de las imágenes; pues, aunque no es el propósito principal de esta investigación, está enfocada a que en un futuro sí que se puedan implementar códigos más complejos de visión artificial en la placa.

La biblioteca de OpenCV (*Open Source Computer Vision,* Visión por Computadora de Código Abierto) es una de las más usadas a nivel mundial, y está dividida en distintos módulos, dedicados a diversos ámbitos de la visión por computadora.

OpenCV trabaja esencialmente con matrices, su objeto principal es de clase *Mat,* y esta matriz contiene los valores de los píxeles de una imagen y otras características de la misma (Ivan, David, Tomislav, Hrvoje, & Mario, 21-25 May 2012).

4.1.2. Concepto de píxel y espacio de color

Puesto que el código que va a ejecutar la Raspberry Pi en su arranque es un conteo de los píxeles de una foto que tome en ese mismo instante, y en OpenCV tratan las imágenes como matrices donde cada dato son los valores del píxel, parece conveniente realizar una breve descripción de este concepto.

La palabra es un acrónimo de *Picture Element*, es decir, elemento de imagen. Se puede definir un píxel como "elemento de color uniforme más pequeño que forma una imagen digital" (Barros, 2016). Este elemento contiene valores de color, saturación y brillo, y su tamaño depende del dispositivo que se use.

Por lo general, se sigue el modelo de color RGB ("*Red, Green, Blue",* Rojo, Verde, Azul), donde cada píxel se define por tres datos y se asignan *n* bits a cada uno, por lo que los valores oscilan entre 0 y 2^{n-1}. Combinando un valor entre 0 y 2^{n-1} en cada uno de los tres componentes RGB es posible obtener todo el espacio de colores.

Sin embargo, para este caso resulta más útil cambiar del espacio RGB a HSV (*Hue, Saturation, Value;* Matiz, Saturación, Valor).

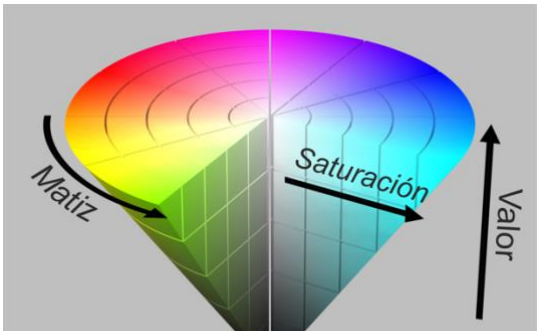


Figura 8: Espacio de colores HSV

Fuente: Maulucioni, 2015

El modelo HSV es una transformación no lineal del modelo RGB en coordenadas cilíndricas, y constituye una representación más cercana a la forma en que los humanos perciben los colores y sus propiedades (EcuRed, 2015).

El motivo por el cual resulta más útil utilizar el espacio de color HSV es porque en RGB cada uno de los tres valores que componen el píxel representa un color en sí. En este caso se necesita un rango de colores, es decir, de verde claro a verde oscuro, por lo que podría dar problemas el hecho de delimitar un rango de tres colores a otros tres. En cambio, en el espacio HSV cada valor representa un elemento independiente del otro, y entre los tres se obtiene el color.

4.1.3. Conteo de píxeles

En este apartado se incluye el pseudocódigo del programa que ejecuta la Raspberry Pi para realizar el conteo del número de píxeles de color verde de la imagen.

1. Inicio
2. Importar bibliotecas
3. Tomar foto con *picamera,* guardándola con el nombre “imagen”
4. Crear la variable “imagen”; siendo “imagen” la foto tomada previamente, importada desde OpenCV
5. Convertir “imagen” del espacio de color RGB a HSV
   hsv = “imagen” en HSV
6. verdes_bajos = límite inferior del rango de verdes
   verdes_altos = límite superior
7. Aplicar máscara sobre el rango “verdes_altos”, “verdes_bajos” en “hsv”, que es la foto tomada convertida al espacio HSV.
   mask = píxeles dentro del rango especificado
8. verde = píxeles de la máscara
9. Fin

La variable “verde” sería el resultado deseado del programa, y se consigue con la función *countNonZero* (contar los “no cero”). Esto es posible gracias a la máscara, que lo que hace es dar valor 2^{n-1} -es decir, color blanco (ver sección 4.1.2)- al rango

que se especifica, y cero al resto. En la Figura 9 se puede observar el funcionamiento de ésta.

Figura 9: Foto tomada por la Raspberry Pi, a la derecha con la máscara aplicada y a la izquierda sin ella.

Fuente: Elaboración Propia

El código completo se encuentra en el Anexo II.

4.1.4. Ejecución automática de la tarea

Finalmente, el paso que falta para completar la programación de las tareas de la Raspberry Pi sería introducir el *script* con el código en el arranque del sistema operativo, de forma que no haya que iniciarlo de forma manual, sino que se ejecute cada vez que se encienda la placa. Esto es primordial, ya que el sistema debe trabajar de forma autónoma.

Algunas opciones que se consideraron para conseguir esto fueron: LXDE, crontab, incluir la dirección del *script* en el fichero */etc/init.d*, o en este otro */etc/rc.local*.

El problema principal encontrado en la realización de esta tarea fue el otorgar permisos de lectura, escritura y ejecución al *script*, ya que esto es imprescindible para poder ejecutar el código en el arranque del sistema.

Se decidió introducir la ruta del *script* deseado en el archivo */etc/rc.local*, pues LXDE resulta más útil para el inicio de un programa en el entorno gráfico, lo cual no interesa en este caso. Respecto a los otros métodos mencionados, se encontró la dificultad de que no ofrecían flexibilidad a la hora de modificar el archivo una vez dados los permisos para su ejecución en el inicio del sistema.

4.2. Configuración Pycom

Puesto que la mayor parte del código que ejecuta el microcontrolador se encarga de establecer la comunicación serial, enviar los datos a la nube o controlar el voltaje que le llega a la Raspberry Pi para su encendido o apagado, en este punto se describirá brevemente el funcionamiento de este dispositivo en líneas generales, sin entrar aún en su código principal.

Como ya se explica en el apartado 3.1, el intérprete utilizado para programar la tarjeta es Atom, junto con el paquete Pymakr. De esta forma, al conectar la tarjeta al puerto USB del ordenador (puerto COM), es posible controlarla desde la consola de Atom.

Tanto para estos pasos iniciales como durante la creación del código que ejecutará el microcontrolador ha sido de mucha utilidad la documentación existente en la página oficial del mismo (Pycom , 2019).

Se denomina “proyecto” a los ficheros que posteriormente cargaremos en la tarjeta. Se creará una carpeta para dicho proyecto, que consta de la siguiente estructura:

- boot.py: se ejecuta una sola vez al encender la tarjeta.
- main.py: como su nombre indica, contiene las líneas principales de código.
- lib: en esta carpeta se pueden incluir otros *scripts* con funciones, bibliotecas, etc.

El único archivo imprescindible del proyecto es el *main.py*, y éste es el único que se usará para este proyecto.

Sin embargo, para que este código se mantenga en la tarjeta, es necesario crear otro archivo de configuración, llamado *pymakr.conf*. En este archivo se especificará la dirección del COM del ordenador al que está conectada la tarjeta.

4.3. Comunicación serial

La comunicación no es bidireccional, es decir, se va a realizar en un único sentido: la Raspberry Pi va a transmitir un dato y la tarjeta Pycom va a recibirlo, pero no necesitamos que esta última envíe nada de vuelta. Esto ocurre porque es la tarjeta Pycom la que administra la comunicación, espera el dato y cuando lo recibe desconecta a la Raspberry Pi de la alimentación.

En esta sección se describirá cómo se ha realizado la comunicación a través de puertos serie entre la Raspberry Pi y la placa LoPy, utilizando para ello Python, como ya se ha comentado con anterioridad, y su biblioteca específica para ello: *PySerial.*

Sin embargo, es importante que previamente se mencionen los otros métodos que podrían haber servido para establecer la comunicación entre ambos dispositivos, y las ventajas o beneficios que han llevado a que finalmente se decida utilizar la comunicación serial.

4.3.1. Elección del método de comunicaciones

Los métodos que se plantearon que podrían servir para establecer la comunicación, además de la comunicación serie, fueron:

- Programación de sockets
- Bluetooth
- Protocolo MQTT

Los **sockets** establecen comunicaciones tipo cliente/servidor, que es lo que se necesitaría, en la capa de transporte. Actúan a modo de buzón entre ambos dispositivos, y los dos pueden tanto leer información como enviarla.

La desventaja -para las necesidades que se requieren- de este tipo de comunicación es que invierte tiempo cada vez que se realiza la conexión en iniciarla en ambos dispositivos y en que el cliente envíe una petición de conexión al servidor y éste la acepte. Además, el problema principal es que ambos dispositivos deben estar conectados a la misma red. Para implementar este tipo de comunicación es necesario que los dispositivos estén conectados a través de una red local inalámbrica, generada

por un router WiFi. El router gestiona la red local para permitir la comunicación entre los dispositivos conectados a ella sin necesidad de estar conectado a otra red o a Internet, sin embargo, no deja de ser un inconveniente puesto que supone un coste extra y sería necesario incluir también repetidores debido a la amplitud del terreno a cubrir.

Por otra parte, **Bluetooth** se descartó también, pues aún siendo un método sencillo, consume mayor energía por parte de los dispositivos en comparación con la comunicación serial. Además, al ser una comunicación inalámbrica es más vulnerable en términos de seguridad que la comunicación serial, que se realiza por cable.

Por último, se investigó acerca de **MQTT** *(Message Queue Telemetry Transport),* un protocolo a nivel de las capas superiores del modelo OSI y que normalmente se apoya en TCP/IP. Es muy usado para el Internet of Things, ya que consume muy poco ancho de banda y está orientado a la comunicación de sensores. Se basa en un modelo de suscripción y publicación, con una topología en estrella en la que en el centro se encuentra el *bróker*, y los clientes se conectan a él (IBM, 2018). Se decidió no usar este protocolo por la necesidad de conexión a una red WiFi o cableada (Pycom GitBook, 2019), además de ser un protocolo bastante complejo para la comunicación que se requiere.

Por todo lo mencionado, el método más adecuado para las características que la comunicación que se quiere establecer entre la SBC y el microcontrolador resulta ser la comunicación por puertos serie.

La **comunicación por puerto serie** se realiza en banda base. Por lo general, al dato se le añaden unos bits extra al principio y al final, y uno de paridad para controlar errores en las tramas a nivel de enlace (*start bit, stop bit* y *parity* en el código del Anexo III).

Una comunicación serie genérica requiere tres conexiones (ver Figura 10):

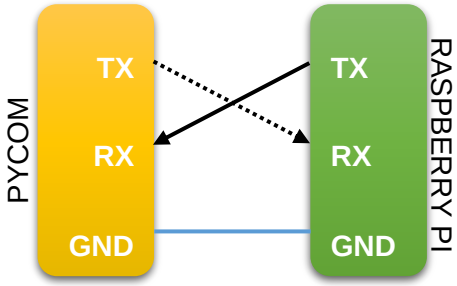


Figura 10: Esquema comunicación serial

Fuente: Elaboración Propia

No obstante, para este caso, como ya se ha explicado no será necesaria la conexión de la flecha punteada de la Figura 10.

Los pines Tx son los de transmisión y los Rx los de recepción. De gestionar la comunicación serie se encarga una interfaz integrada en el microcontrolador que se denomina UART. Esta se puede programar desde Python.

4.3.2. Implementación de la comunicación serial

Al escoger los pines de la UART Tx /Rx (ver Figura 10) de la Raspberry Pi es importante tener en cuenta el modelo, ya que en anteriores modelos que no tenían Bluetooth había dos UART, y ahora solamente hay una, pues la otra se utiliza para gestionar la comunicación Bluetooth.

Se utilizarán los pines señalados en amarillo en la Figura 11, en concreto el pin 14, que es el que se encarga de transmitir (Tx), y la masa (*Ground*), señalada con el color negro. Para ver el mapa completo de pines, ir a Figura 3.

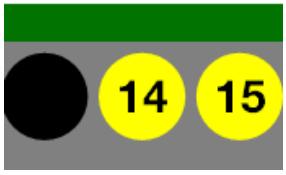


Figura 11: Pines escogidos de la Raspberry Pi

Respecto al microcontrolador Pycom, los pines predeterminados para la comunicación serie son Rx1/Tx1, que se corresponden con P3 y P4 en la Figura 5. Sin embargo, es posible configurar cualquier otro pin para cumplir este propósito.

Se encontró el problema de que al conectar la placa al PC a través del USB no era posible cargar los programas a la tarjeta ni depurar los códigos, pues se desconfiguraba al tratar de probar el programa a través de la consola de Atom. Esto ocurre porque la conexión con el PC se realiza por puerto serie también y emplea los pines P3 y P4, imposibilitando su uso en la fase de depurado.

Se corrigió este problema configurando otros pines para la comunicación, como puede verse en el código del Anexo III, en la línea:

uart1 = UART(1, 115200, bits=8, parity=None, stop=1, pins=(None,'P21'))

En concreto, se utilizó el pin P21 como Rx. Se muestra una vista ampliada de la ubicación del mismo en la figura siguiente.

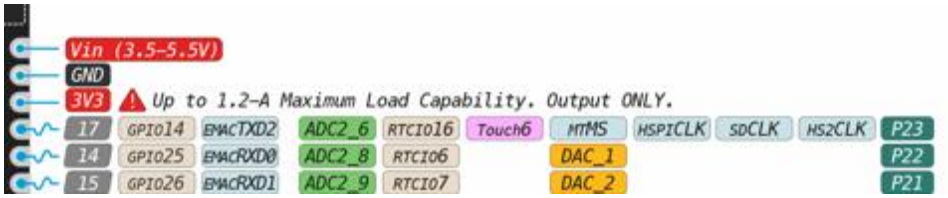


Figura 12: Pines escogidos del microcontrolador

Los códigos completos que ejecutan tanto la Raspberry Pi como el microcontrolador Pycom se encuentran en los Anexos II y III respectivamente.

4.4. The Things Network

4.4.1. Iniciativa y funcionamiento

The Things Network o TTN es una iniciativa que comienza en Amsterdam en el año 2015, cuyo objetivo es construir una red global de IoT a la que cualquier persona pueda unirse y contribuir. Se basa en las “comunidades”, es decir, grupos de voluntarios en todo el mundo que trabajan para desplegar la red en su área o ciudad. Las redes establecidas por TTN son de carácter público, de tal manera que cualquier ciudadano puede conectar sus dispositivos y recolectar información. La idea es que esto contribuya a la mejora de la calidad de vida en las ciudades (Salvador, 2018).

Actualmente existen ya más de 8000 *gateways* conectados y funcionando en todo el mundo (The Things Network Community, 2019). A continuación, en la Figura 13 se puede apreciar su distribución en un mapa.

Figura 13: Distribución mundial gateways y comunidades TTN

The Things Network eligió como tecnología de comunicación LoRaWAN sobre LoRa, por las ventajas que dicha tecnología presenta (ver sección 2.2) en lo referente a alcance, consumo de batería y coste (Briceño, 2018).

Teniendo en cuenta la arquitectura de una red LoRaWAN, Figura 14, The Things Network representa el servidor red al que llegan los datos desde los *gateways* o pasarelas. Posteriormente, para el tratamiento y procesamiento de dichos datos se integrará con una aplicación acorde a lo que se quiera realizar. En este caso, simplemente se ha integrado con una plataforma que almacena la información que llega y la mantiene disponible durante siete días. Se profundizará sobre este tema en la sección 4.4.6.

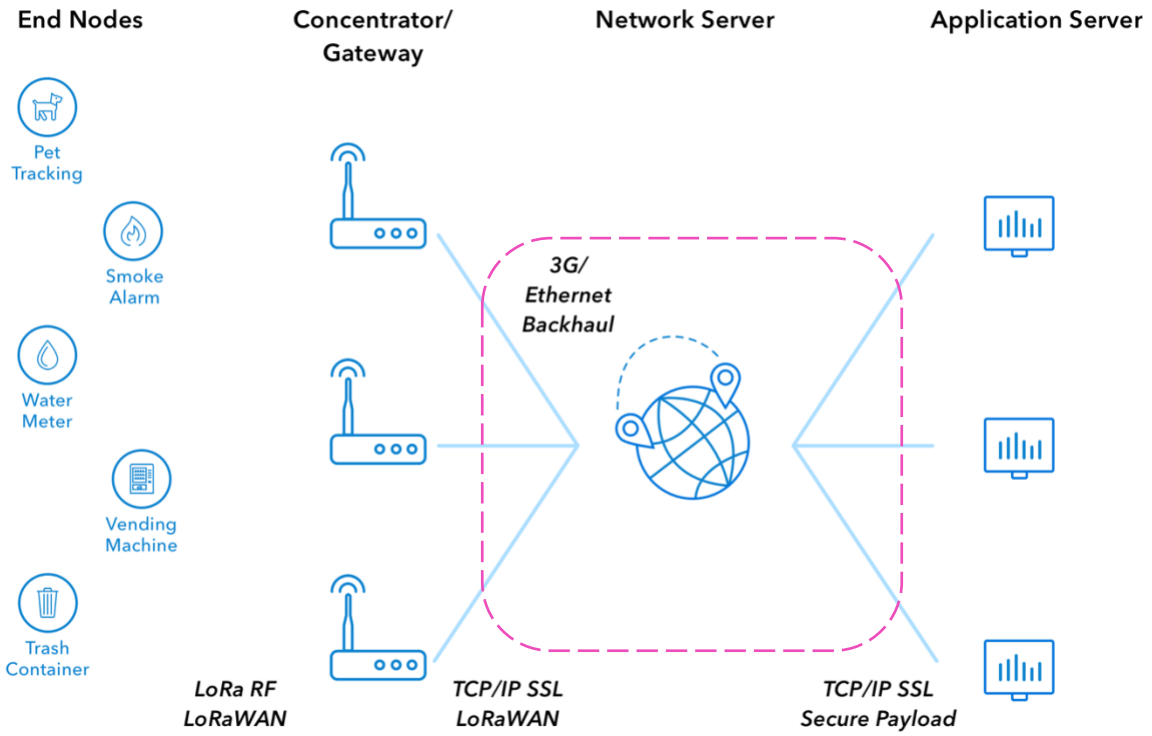


Figura 14: Colocación de antenas gateway

Fuente: Elaboración Propia

4.4.2. Configuración del gateway

Como se definió en el apartado de hardware utilizado, se ha hecho uso del *gateway* Sentrius RG1xx de Laird. Para su configuración inicial se siguieron los pasos de la guía del fabricante, versión 2.1 (Laird, 2018).

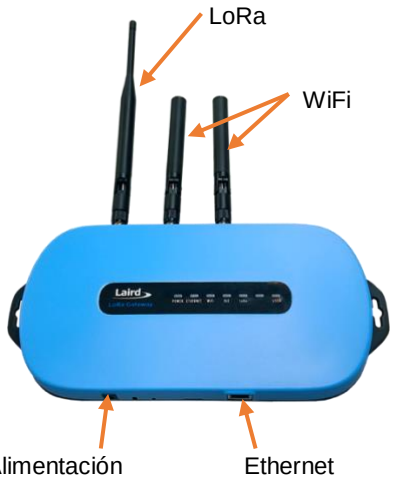


Figura 15: Arquitectura de red LoRaWAN

Fuente: LoRa Alliance, 2019

Una vez colocadas las antenas como se indica en la Figura 15, la fuente de alimentación y cable de Ethernet (IEEE 802.3), se puede acceder desde un navegador a la interfaz web del dispositivo. Desde dicha interfaz será desde donde se configure tanto la conexión WiFi como LoRa del *gateway*.

En la configuración de LoRa es importante elegir la opción “The Things Network Legacy”. Tras realizar la configuración del dispositivo en su interfaz web, será necesario crear una cuenta en The Things Network y registrar el *gateway* en ella.

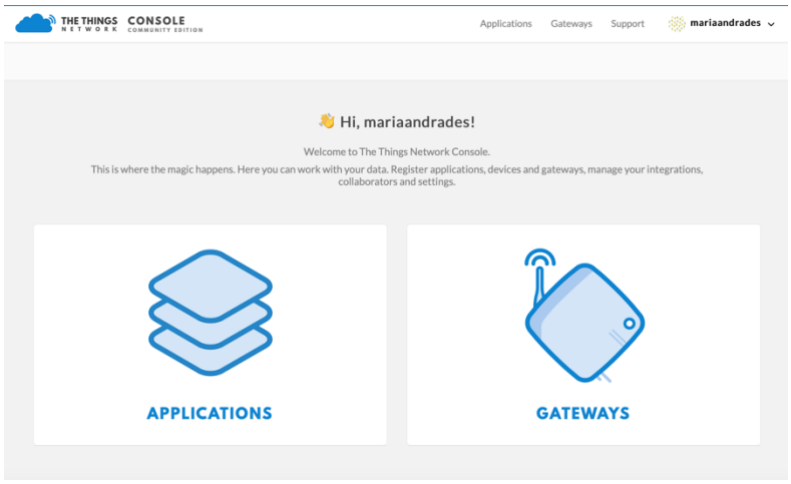


Figura 16: Consola de The Things Network

Dentro de la cuenta creada en la web de la comunidad, existen dos opciones, como se aprecia en la Figura 16: Aplicaciones y Gateways.

Dentro de la segunda opción es donde se registrará el *gateway*, utilizando para ello el ID de su reverso y cierta información como descripción o plan de frecuencia, en este caso, Europa 868 MHz.

En la Figura 17, la configuración en TTN del *gateway* utilizado para el presente proyecto.

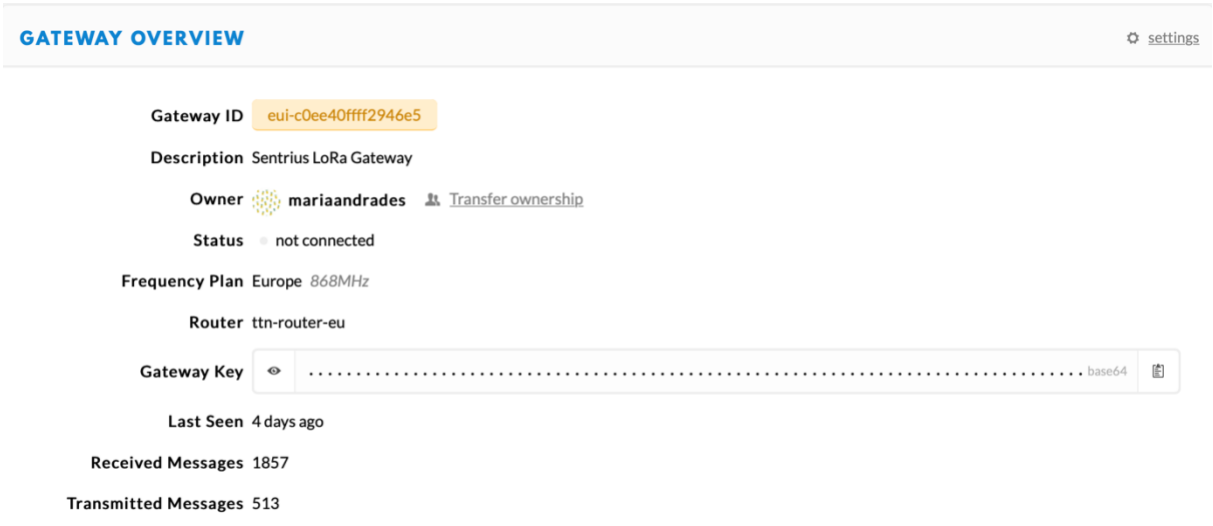


Figura 17: Registro del gateway en TTN

4.4.3. Creación de aplicación y registro de nodos

Para poder gestionar los datos que se reciban de los nodos y su posterior procesado y almacenamiento, es necesario crear una aplicación para este fin. TTN da la posibilidad de crear varias aplicaciones utilizando un mismo *gateway*, gestionadas desde una misma cuenta de usuario.

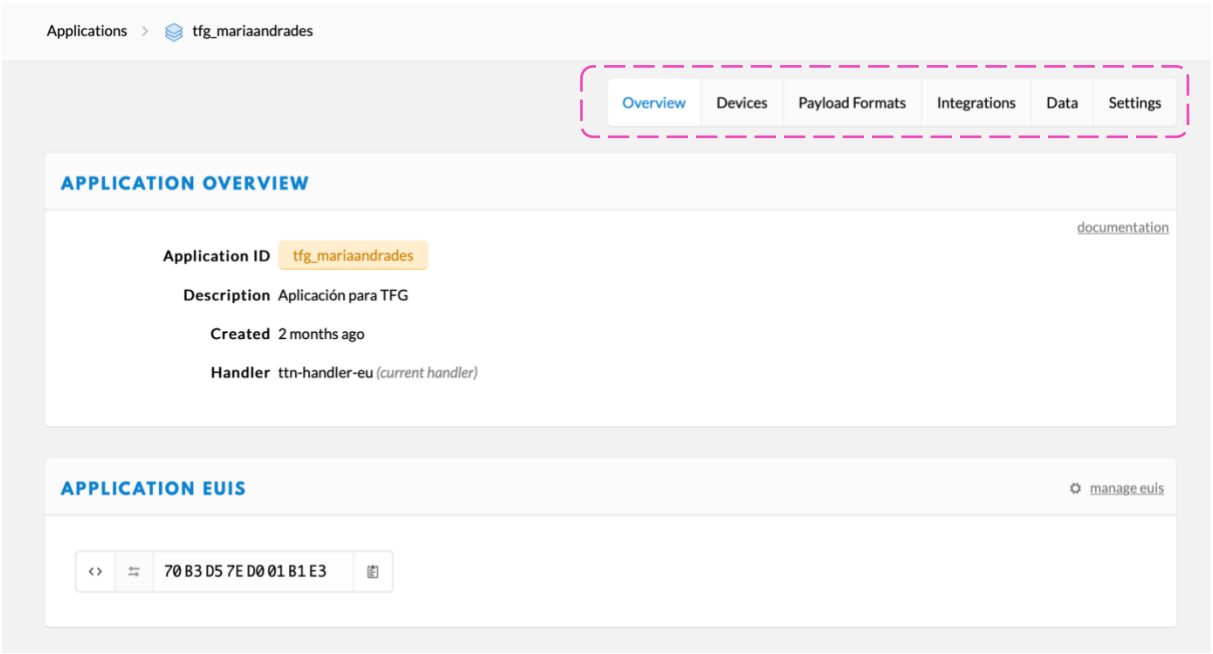


Figura 18: Creación de aplicación en TTN

De manera similar a como se registró el *gateway*, se añadirá una aplicación. Será generada una clave de acceso y un identificador único de la aplicación (EUI).

En la esquina superior derecha de la Figura 18, se encuentran las distintas opciones que se pueden configurar en la aplicación. Se describe a continuación para qué sirve cada una de las pestañas o qué datos encontramos en ellas.

En primer lugar, se encuentra ***Overview***, esta es la pestaña que se muestra en la figura anterior (18) y en ella se encuentran datos generales acerca de la aplicación, es decir, identificador, descripción, claves, dispositivos finales o nodos registrados, colaboradores, etcétera.

En la segunda pestaña, ***Devices***, es posible registrar y administrar los distintos dispositivos que vayan a enviar información a la aplicación. Se explicará en profundidad en la sección siguiente cómo se ha hecho.

En ***Payload Formats*** es posible personalizar tanto el formato de los mensajes que llegan desde el dispositivo final como los que se devuelven a este en forma de ACK o confirmación. En este caso se han decodificado los mensajes recibidos (ver apartado 4.4.5).

Integrations es la sección en la cual se pueden conectar los dispositivos a una aplicación externa, de forma que el usuario no tenga que utilizar la consola de The Things Network ni sincronizar datos continuamente. Las integraciones se encargan de vincular The Things Network con una plataforma de IoT externa donde el usuario puede administrar las aplicaciones y los dispositivos, mientras que el proceso de integración se encarga de sincronizar con TTN.

A continuación, está la pestaña de ***Data***, en ella será posible ver en tiempo real cómo van llegando los mensajes de los distintos dispositivos conectados a la aplicación. Además de estos mensajes, llamados *uplink messages*, se pueden ver las señales de activación de los nodos, los mensajes en sentido descendente o *downlink messages*- es decir, los que se devuelven al nodo- los mensajes de ACK o acuse de recibo y los de error.

Finalmente se encuentra la pestaña de ***Settings***, donde es posible cambiar las claves que se establecen por defecto, añadir colaboradores, etc.

4.4.4. Configuración del nodo

4.4.4.1. Configuración en TTN

Para esta aplicación se ha configurado un único nodo, al que se ha llamado “pycom”, como se observa en la Figura 19. Dentro de la configuración del dispositivo se pueden ver los datos enviados por ese nodo en concreto, en la pestaña *Data* de la esquina superior derecha.

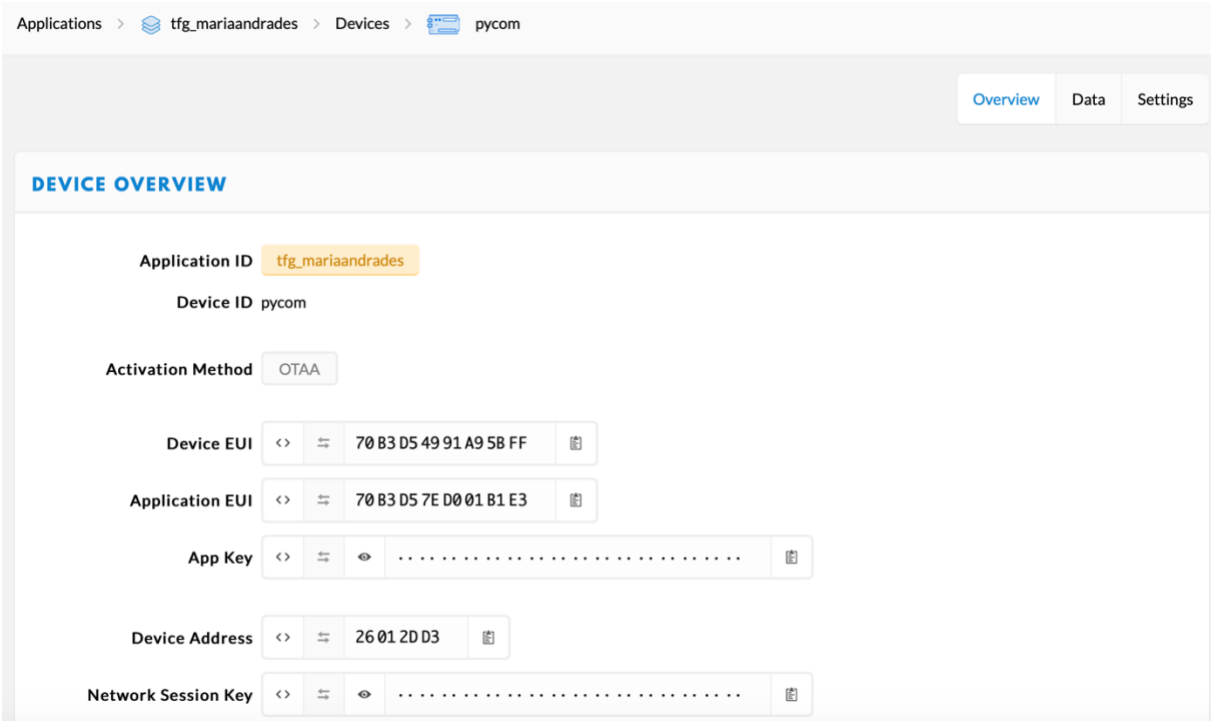


Figura 19: Configuración de un nodo TTN

Para conectar el nodo a la red LoRaWAN existen dos opciones: OTAA (*Over The Air Activation* o Activación en el aire) y ABP (*Activation By Personalization* o Activación Por Personalización). Se explica a continuación el funcionamiento de cada una de ellas y cuál se ha escogido.

En cualquier caso, es una opción modificable desde la configuración del dispositivo en el servidor de TTN, siendo necesario cambiar también el código del microcontrolador. No es necesario utilizar siempre el mismo, diferentes nodos pueden utilizar diferentes métodos de activación dentro de una misma aplicación del servidor.

Por un lado, **OTAA** supone la opción más segura de unión a la red LoRaWAN. Utiliza tres parámetros para su configuración:

- DevEUI: identificador único del dispositivo.
- AppEUI: identificador de la aplicación.
- AppKey: clave de 16 bytes que utilizan el nodo y la red para establecer las claves de sesión.

Para establecer la conexión, el nodo solicita un inicio de sesión con estos parámetros, esta solicitud llega al servidor a través del *gateway*, y el servidor verifica que el nodo esté registrado y que las claves sean correctas. Solo si esto es así, se asigna una sesión temporal al nodo y se le envía. Una vez el nodo la recibe, puede enviar datos a la red. La principal ventaja de este método es la seguridad, puesto que las claves de sesión se renuevan cada vez que se inicia una.

Por otro lado, se encuentra el método de activación **ABP**. Éste es un método de conexión más sencillo, pues las claves y parámetros de configuración están predefinidos tanto en la red como en el dispositivo. El dispositivo envía los datos usando dichos parámetros, el *gateway* comprueba si son correctos, si lo son se procesan y en caso contrario se rechazan. Los parámetros de configuración de este modo de conexión son:

- DevAdress: equivalente a una dirección IP del dispositivo.
- NetworkSessionKey: clave de red
- ApplicationSessionKey: tanto este parámetro como el anterior son claves generadas al registrar el nodo con este modo.

La ventaja principal del modo ABP es que el nodo no requiere confirmación para enviar mensajes a la red. Esto puede ser útil con dispositivos en movimiento, por ejemplo.

El método escogido fue OTAA, por las ventajas que aporta en lo referente a seguridad, según se ha explicado con anterioridad.

4.4.4.2. Programación del dispositivo

Se incluye en este subapartado un pseudocódigo de la parte del programa del microcontrolador encargada de establecer la comunicación LoRa con la red. El código completo se encuentra en el Anexo III.

1. Inicio
2. Importar bibliotecas
3. Inicializar modo LoRaWAN como *lora*
4. Crear variables *app_eui* y *app_key*, parámetros de configuración para la conexión con el método OTAA
5. Enviar petición de inicio de sesión
6. Mientras no se haya unido a la red
   Dormir 2.5 segundos
7. Crear variable s = socket de LoRa
8. Bloquear socket hasta que se envíe el dato
9. Empaquetar variable data, que es la que se va a enviar
10. Enviar dato empaquetado
11. Fin

4.4.5. Decodificación

Los datos llegan en forma de bytes, como se ve en la Figura 20, el campo “payload”.

Figura 20: Llegada de datos en TTN

Para conseguir el dato en base decimal e indicar lo que representa, como en el ejemplo de la figura previa, que se especifica “pixeles: 4577”, es necesario introducir un código JavaScript (JavaScript, 2019) en el apartado de *Payload Formats* de la aplicación.

El motivo por el que se ha codificado o empaquetado el dato antes de enviarlo en el microcontrolador -ver apartado 4.4.4.2- y posteriormente es necesario decodificarlo en este punto es porque para usar de forma efectiva la red LoRaWAN es preciso mantener los paquetes transmitidos lo más pequeño posible (Core Electronics, 2019).

En la siguiente línea del código que ejecuta el microcontrolador para enviar el dato, puede observarse que el formato que se asigna es ‘q’. Esto implica que la variable es entera y tipo “long long”, ya que en este caso podría ser un número del orden de un millón. Por tanto, añadiendo este formato se envía un paquete de 8 bytes. Con 8 bytes se podrían codificar números de 0 a $(2^8)^8$.

datapack=ustruct.pack('q', data)

El código de JavaScript se basa en (ver Anexo IV):

1. Inicio
2. Llamar a la función *Decoder*, que lee *bytes* y *port* (paquete de bytes recibido y puerto de entrada)
3. Declarar el objeto *decoded*, que será el resultado que se devuelva
4. Declarar el objeto *entero*, en el que se decodifican los bytes
5. Se la asigna el nombre *pixeles* a la variable
6. Devuelve el resultado
7. Fin

4.4.6. Integración

En el servidor de The Things Network es posible ver en tiempo real los datos que están enviando los nodos, sin embargo, no es posible verlos más tarde si no se tiene la web del servidor abierta, es decir, no se guardan. Además, al refrescar la web del servidor, se eliminan los datos recibidos con anterioridad. Es útil, por tanto, almacenarlos en algún lugar para revisarlos o analizarlos más tarde.

Una forma sencilla de realizar esto es con una integración. Como se ha explicado antes, la integración se encarga de sincronizar los dispositivos con The Things Network, y el usuario puede acceder a los datos enviados por dichos dispositivos en cualquier momento, sin necesidad de estar conectado a la web del servidor.

Se ha realizado la integración con la plataforma Data Storage v2.0.1, de The Things Industries B.V. (The Things Industries, 2019). En esta plataforma se almacenan los datos y los pone a disposición del usuario a través de una API (*Application Programming* Interface o Interfaz de Programación de Aplicaciones). Los datos se almacenan por un período de hasta siete días.

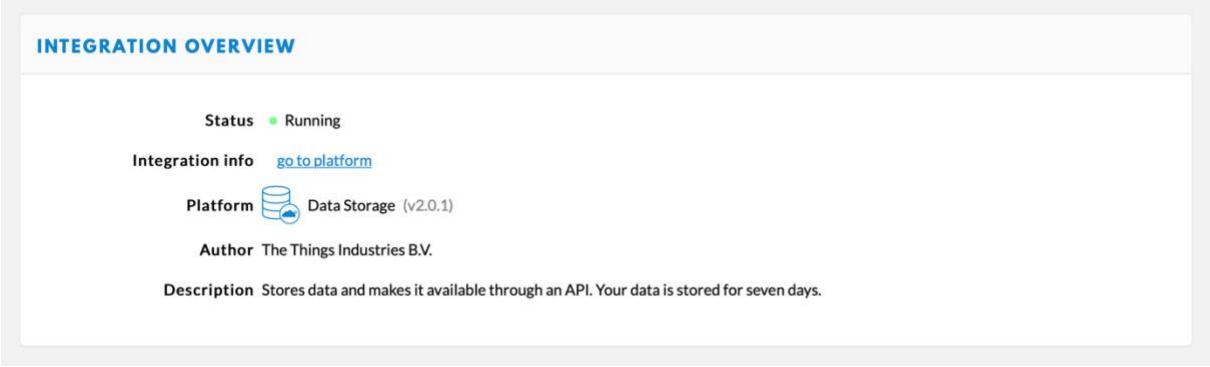


Figura 21: Integración en TTN

En la opción “go to platform”, en la Figura 21, se accede a la web donde se encuentran los datos de la aplicación (Swagger UI). Una vez se accede a la URL es necesario dar autorización para mostrar los datos, introduciendo la clave de la aplicación correspondiente.

4.5. Control del encendido y apagado de Raspberry Pi

4.5.1. Funcionamiento

Hasta ahora se ha conseguido que la Raspberry Pi tome la foto y realice ciertos cálculos sobre ella, que se la envíe al microcontrolador, y que éste la envíe usando la tecnología LoRa a través de una red LoRaWAN al servidor red de The Things Netwotk, a través de un *gateway* LoRa. Una vez allí, se almacena la información en otra plataforma durante siete días gracias al proceso de integración.

Solamente faltaría programar en el microcontrolador cuándo deben ejecutarse estas tareas y cuándo debe permanecer en reposo, y lo más importante, cuándo debe encender o apagar la Raspberry Pi.

El diagrama de flujo del proceso completo sería el siguiente -ver Figura 22- a la izquierda el proceso que ejecuta el microcontrolador y a la derecha el de la Raspberry Pi. Se indica con una línea punteada la parte del bucle de Pycom en el que la Raspberry Pi se encuentra trabajando.

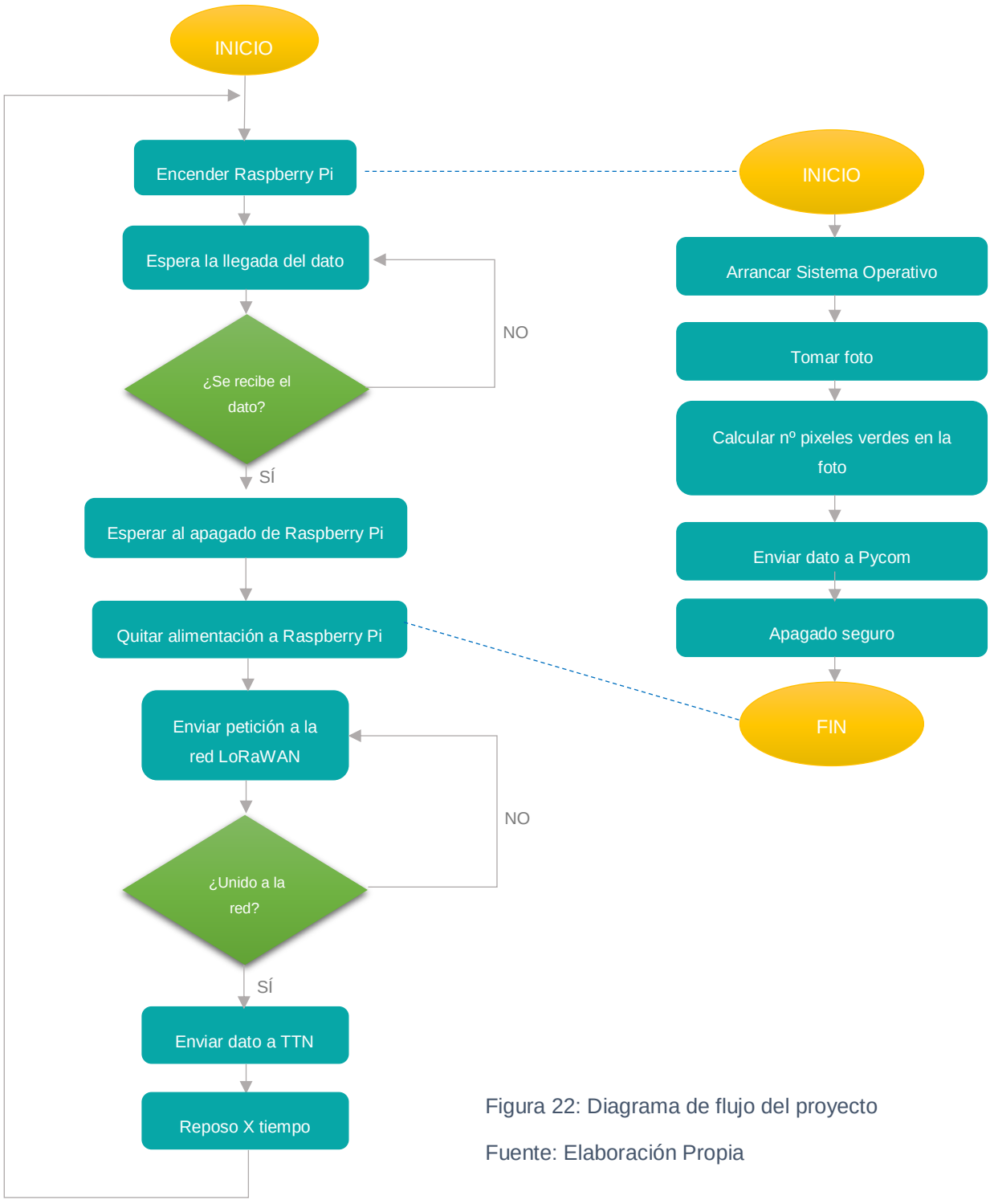


Figura 22: Diagrama de flujo del proyecto

Fuente: Elaboración Propia

Por tanto, se hará depender la alimentación de la Raspberry Pi del microcontrolador. El microcontrolador se alimenta de una pila o batería, y a través de un relé electrónico decide en qué momento del proceso se enciende o se apaga la SBC.

Un requisito importante en el momento de modelar la solución son las necesidades de alimentación de la placa. Volviendo a la Tabla 4 de especificaciones técnicas de la misma, se puede comprobar que necesita 5 V y 2.5 mA, por lo que la batería debe ser capaz de suministrarlo.

Por otro lado, se comprueba que el voltaje de salida que proporciona el microcontrolador es de alrededor de 3.3 V, por lo que el relé debe ser capaz de activarse a esta tensión.

Con estas condiciones, se decide utilizar un relé electromagnético *finder 40.31* de activación a 3 V DC, como el de la Figura 23.

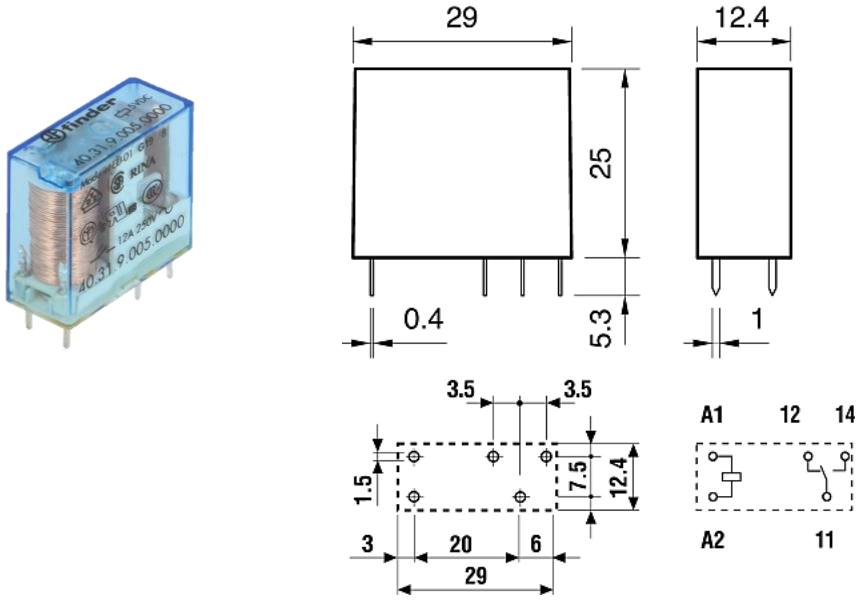


Figura 23: Relé utilizado

Fuente: RS Components Spain

En las patillas A1 y 12, según la Figura 23, se conectaría la tierra y en la A2 la salida del Pycom, mientras que en la patilla 11 se conectaría la batería, es decir, 5 V. La patilla restante, 14, sería la salida del relé y, por tanto, la alimentación de la Raspberry Pi. De esta forma, si V_{out} del microcontrolador es distinta de cero, se activaría el relé y la Raspberry Pi se alimentaría con la batería.

4.5.2. Esquema de conexiones

En esta sección se muestra un esquema de las conexiones eléctricas realizadas para hacer depender la alimentación de la Raspberry Pi del Pycom, y la alimentación de éste, a su vez, de una batería.

Como se menciona en la sección anterior, se utiliza un relé para el control de dicha alimentación, sin embargo, se observa en la práctica que, aunque el relé se activa a 3 V y del pin de salida del microcontrolador se obtienen 3.3 V, la intensidad no es suficiente para activarlo.

Este problema se solventó utilizando un integrado L293D, de forma que la salida del Pycom se utiliza para activar el *driver* del integrado. Éste último se alimenta con la batería. En la Figura 24 viene representado un esquema de la conexión y en la 25, el *datasheet* del integrado L293D.

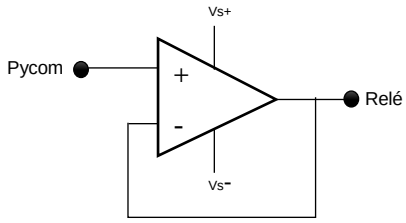


Figura 24: Esquema de conexión del relé

Fuente: Elaboración Propia

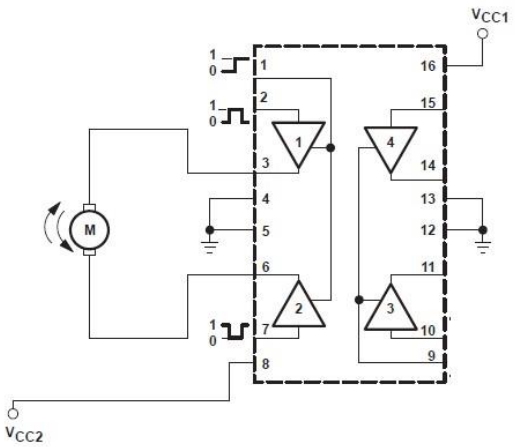


Figura 25: Datasheet integrado L293D

Fuente: Kit Electrónica, 2017

El esquema de las conexiones establecidas para la integración de los dispositivos se presenta a continuación, en la Figura 26. Se ha dibujado una pila para representar la alimentación. En realidad, se ha utilizado un *power bank* como fuente de alimentación del conjunto.

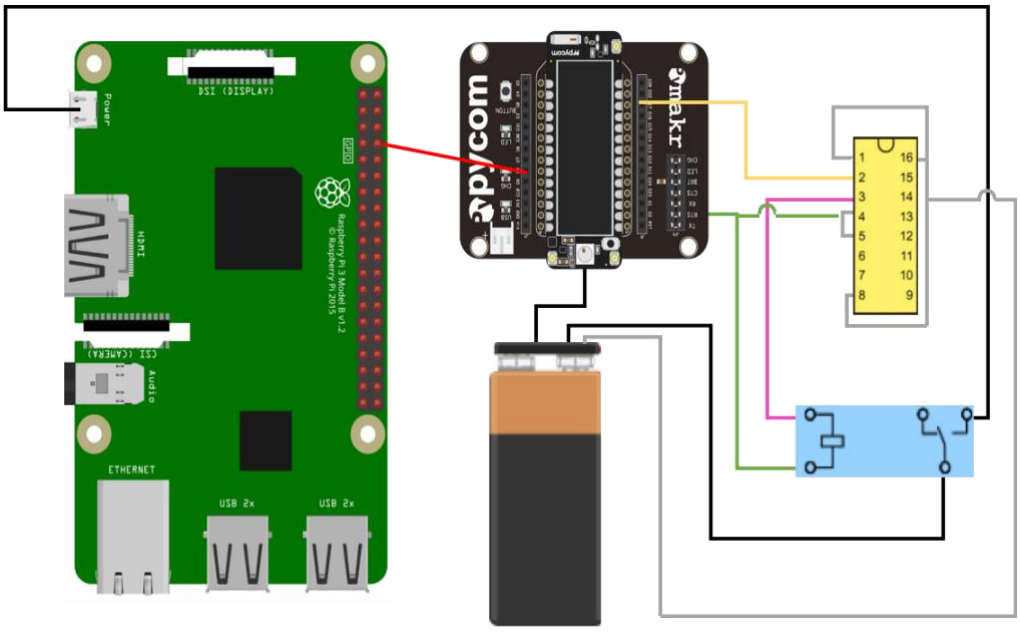


Figura 26: Esquema de conexiones de la integración completa

Fuente: Elaboración Propia

5. Resultados

En este capítulo se pretende cuantificar el ahorro en términos energéticos que supone la integración de ambos dispositivos. Para ello, se emplean los datos teóricos sobre consumo energético de ambas placas, es por tanto un cálculo aproximado.

Según los datos de Raspberry Pi Dramble (Geerling, 2018), la Raspberry Pi 3 Modelo B consume entre 230 mA en modo de bajo consumo hasta 730 mA en pleno rendimiento. Dado que en este caso no se encuentra funcionando a pleno rendimiento: no se usa WiFi, Bluetooth… se ha empleado para los cálculos un valor aproximado de 480 mA, a 5 V y funcionando una media de 54 segundos cada vez que se ejecuta la tarea.

El microcontrolador, por su parte, se alimenta a 3.3 V y consume 108 mA cuando está utilizando LoRa, 34.5 mA ejecutando el programa y alrededor de 20 mA en modo *sleep* (ver datos en Tabla 1). Hay que mencionar que no se ha utilizado el modo *deepsleep*, en caso de hacerse, el ahorro energético sería mayor, pues en este caso se puede reducir el consumo hasta ~20 uA.

El tiempo que el microcontrolador Pycom se encuentra utilizando LoRa en un ciclo es de 15 segundos aproximadamente y 54 segundos realizando el resto de las tareas. El tiempo que pasa en modo *sleep* se ha calculado restando al total de segundos de un día, los segundos que pasa ejecutando el programa o enviando datos a través de LoRa. Dependiendo del número de veces que se realice este proceso a lo largo del día, cambiará este valor.

Para obtener los valores en Julios, se ha realizado la siguiente conversión:

$$[J] = \frac{[V] \cdot [mA] \cdot [s]}{1000}$$

En la Figura 27 se pueden ver representados en un gráfico de barras los resultados de las operaciones previamente mencionadas, en función del número de veces que se ejecute el ciclo al día.

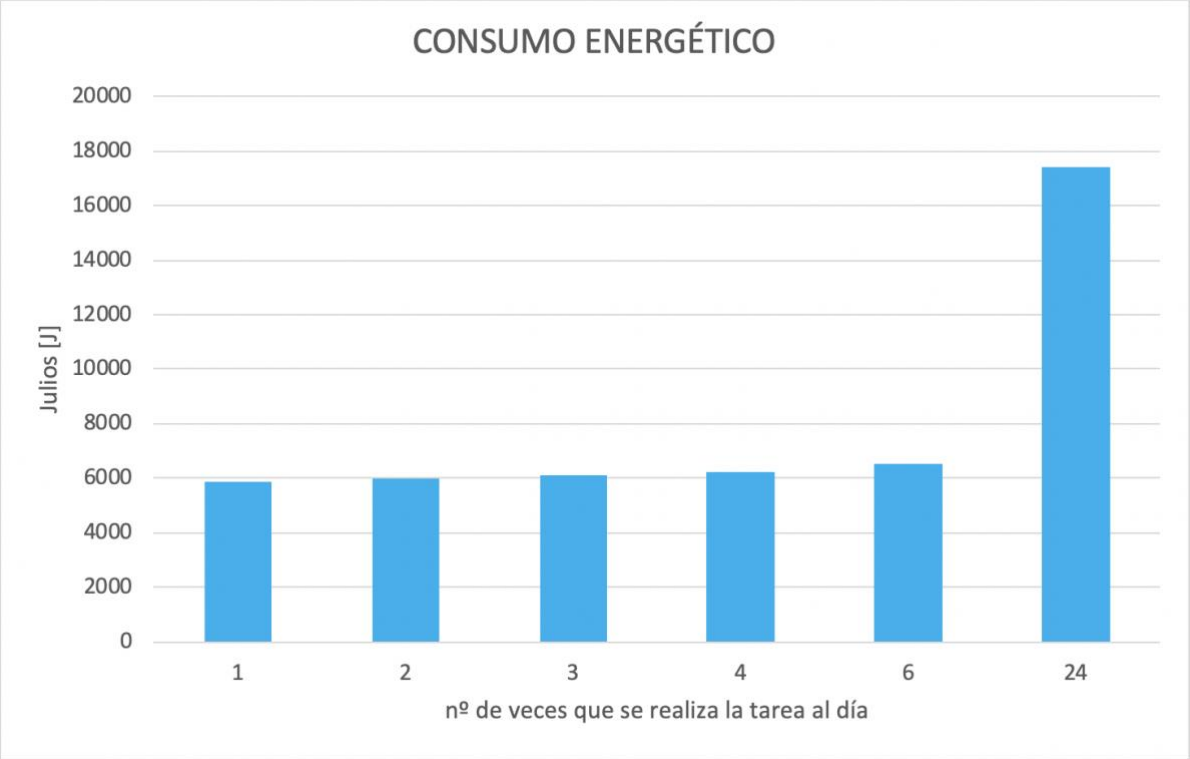


Figura 27: Consumo energético aproximado

Fuente: Elaboración Propia

En la Tabla 7 se ha realizado la comparación de los resultados representados anteriormente en la Figura 27 con los que se habrían obtenido en las mismas circunstancias si no se apagara la Raspberry Pi cada vez que termina de ejecutar la tarea que le corresponde.

	1	2	3	4	6	24
Con apagado [J]	5838,94	5975,48	6112,02	6248,56	6521,64	17388,96
Sin apagado [J]	213069,34	213076,28	213083,22	213090,16	213104,04	213228,96

Tabla 7: Comparación de los resultados con/sin apagado de la Raspberry Pi

Fuente: Elaboración Propia

La comparación gráfica de los resultados presentado en la Tabla 7 se encuentra en la Figura 28.

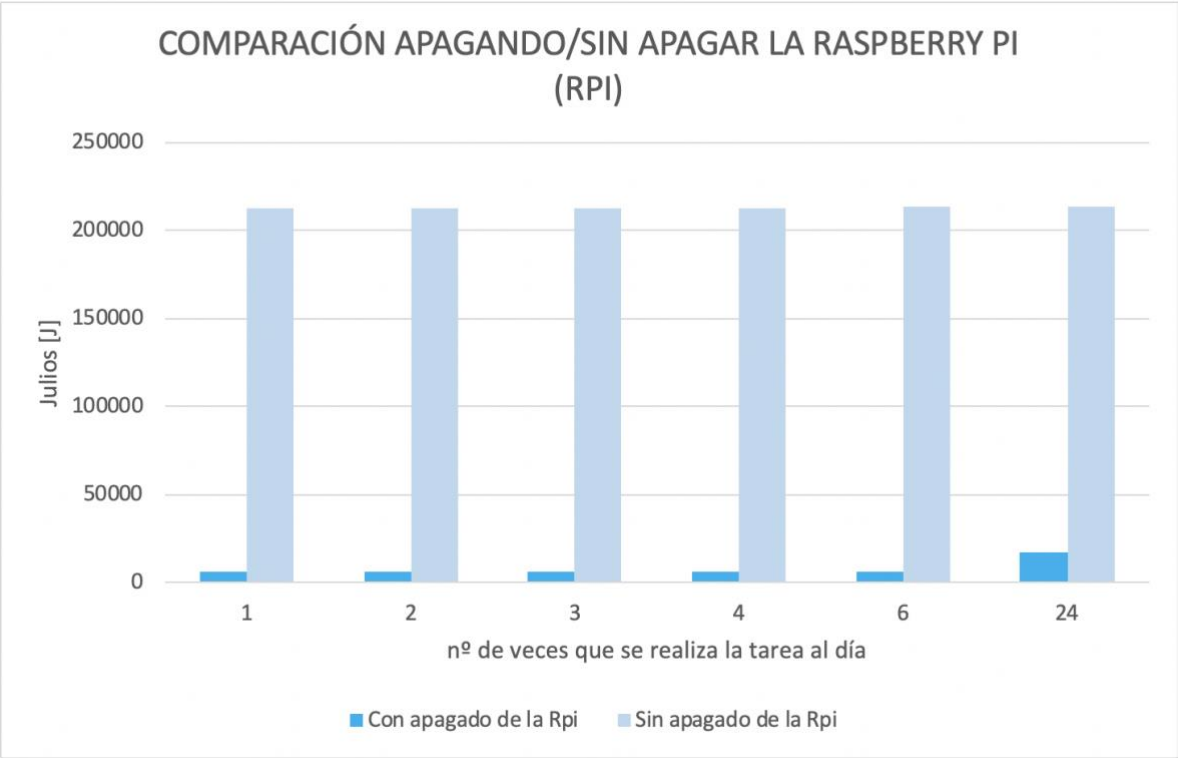


Figura 28: Comparación de los resultados con/sin apagado de la Raspberry Pi

Fuente: Elaboración Propia

6. Posibles aplicaciones

Esta aplicación concreta ha sido diseñada para su uso en el sector agrícola, concretamente en una serie de invernaderos pertenecientes a Surexport. Sin embargo, no tiene por qué limitarse solo a esto, puesto que la investigación es muy general sobre dispositivos para el Internet of Things que sean capaces de cubrir grandes distancias, consumir poco y tener un coste reducido.

6.1. Instalaciones de Surexport

Como ya se ha comentado con anterioridad, esta investigación se realiza como parte del proyecto “INNORIEGA”, financiado por el Instituto de Investigación y Formación Agraria y Pesquera de la Consejería de Agricultura, Ganadería, Pesca y Desarrollo Sostenible (IFAPA).

Figura 29: Invernadero de Surexport S.L.

En la actualidad esta entidad realiza una serie de estudios sobre las necesidades de riego de los cultivos en 32 invernaderos de la empresa Surexport S.L. La mitad de ellos pertenece al cultivo de arándanos y la otra mitad de frambuesas. Se aplican cuatro tipos de riego en los cultivos, en función de la cantidad de agua suministrada: el teórico, por encima y por debajo del teórico necesario, y el que se aplica al resto de cultivos de la empresa, que es mucho más del teórico. Este experimento se repite cuatro veces para los arándanos y cuatro para las frambuesas. Se realiza un seguimiento del crecimiento de los cultivos y la producción en función del riego aplicado, pretendiendo demostrar que las necesidades de riego de las plantas son mucho menores de lo que actualmente se les suministra.

Cada cierto tiempo, los investigadores se desplazan hasta donde se encuentran las instalaciones de la empresa y realizan una serie de fotos para determinar el crecimiento del cultivo en cada uno de los cuatro tipos de riego suministrados. Estas fotografías intentan ser lo más precisas dentro de lo posible, sin embargo, el posicionamiento de la cámara en sitios diferentes cada vez, y la escasa altura de los techos, que hace difícil realizar la foto sin que exista distorsión, hacen que este proceso no sea del todo preciso. Por otra parte, no se utilizan algoritmos de visión artificial, sino que se utiliza un marco con unas medidas conocidas, y a partir de ahí se calcula el porcentaje de verde foto por foto. En este sentido, sería mucho más rápido y cómodo que exista una integración Raspberry Pi-Pycom en el techo de cada uno de los invernaderos que realice una foto de forma periódica, realice sobre ella los cálculos necesarios y envíe los resultados a la nube. De esta forma, los investigadores tendrían un muestreo mucho más amplio del que actualmente se tiene (pues no se desplazan todas las semanas ni mucho menos todos los días a realizar las fotos), además podrían disponer de estos datos desde cualquier lugar, simplemente se necesita un dispositivo con conexión a Internet para ver lo que están enviando los nodos a la nube. Por otro lado, podría implementarse en la Raspberry Pi un programa que calcule el volumen del cultivo de forma mucho más precisa a como se realiza actualmente, o incluso obtener las necesidades de riego dependiendo de ese volumen, de forma que no sea necesario realizar ningún cálculo posterior.

En contraposición, como desventaja o consecuencia negativa de la implantación de esta tecnología, podría mencionarse el coste a asumir. Sería un coste relativamente

alto, aunque a largo plazo podría ser rentable, tanto en términos económicos como de tiempo. Es decir, monitorear 32 cultivos puede ser algo factible, sin embargo, cuando haya que hacer los mismo en un número mucho más elevado de cultivos, incluso que no estén en la misma zona geográfica, la solución que aquí se propone sería mucho más acertada.

6.2. Otras aplicaciones

Este proyecto no deja de ser una base para el IoT, en él se presentan principalmente las comunicaciones entre los dispositivos y de éstos con la nube. La diferencia la marca el programa que ejecute la Raspberry Pi, y en este sentido las aplicaciones pueden ser muy diversas, debido a la gran capacidad computacional de la placa.

Algunos ejemplos de aplicaciones del Internet of Things en las que podría ser útil la integración de Raspberry Pi y Pycom serían:

- Sistema de visión artificial en el sector industrial.

Un sistema de visión artificial instalado en los carros de herramientas de una línea de montaje sería útil para indicar al operario qué herramienta utilizar en cada paso del montaje, además de detectar cuándo ésta se coge y se devuelve al carro. De esta forma se puede saber cuándo se pierde una herramienta y quién, además de mejorar la eficiencia de la línea de montaje y reducir los errores derivados de utilizar la herramienta incorrecta. En grandes fábricas con multitud de herramientas de precio elevado es una gran pérdida económica el extravío de dichas herramientas, por lo que se recurre a soluciones de este tipo.

- Gestión de flotas.

La aplicación del Internet de las Cosas a la gestión de flotas favorece la geolocalización (y con ella el monitoreo de las rutas y la identificación de los trayectos más eficientes), el análisis de rendimiento, el control de la telemetría y el ahorro de combustible, la reducción de emisiones contaminantes al medio ambiente e incluso puede aportar información valiosa para mejorar la conducción de los vehículos (Fracttal, 2019).

- Ciudades inteligentes.

Debido al amplio alcance de las comunicaciones a través de redes LoRaWAN, es posible emplear The Things Network en las ciudades para realizar aplicaciones de interés público o privado, que puedan enviar alertas a los ciudadanos, medir niveles de contaminación, sensores de estacionamiento, iluminación inteligente, etc.

7. Conclusiones

7.1. Consecución de los objetivos

En este proyecto se ha llevado a cabo una investigación sobre la integración de una Raspberry Pi 3 Modelo B y un microcontrolador de Pycom formado por una placa de desarrollo LoPy 4 y una placa de expansión. El objetivo principal era que estos dispositivos realicen una tarea compleja con un consumo energético reducido y desconectados de la red eléctrica.

La integración se ha conseguido de forma exitosa, de forma que el principal objetivo del proyecto ha sido conseguido. Ésta funciona con una batería, por tanto, desconectada de la red eléctrica y en el apartado “Resultados” se puede comprobar cómo el consumo de la integración es mucho más bajo de lo que sería si no se apagara la Raspberry Pi.

En un principio se planteó realizar una monitorización del consumo energético en diversas situaciones, sin embargo, finalmente esto no ha sido posible, pero se ha realizado de forma teórica con los datos sobre el consumo de ambos dispositivos ofrecidos por las respectivas páginas oficiales éstos.

Respecto a los objetivos específicos establecidos en el primer apartado de este proyecto, todos han sido conseguidos. Puntualizar, sin embargo, que siempre es posible mejorar, optimizar aún más el sistema o buscar otras vías de comunicación.

Hay que mencionar también que se han realizado tareas que no se plantearon en un principio, como, por ejemplo, la realización de una placa de PCB, es decir, de circuito impreso, para reducir al máximo el tamaño de la integración de los dispositivos. En ella se encuentra el circuito del relé con el integrado L293D.

Asimismo, se ha realizado también una carcasa para el conjunto. Ésta ha sido diseñada en SolidWorks (SolidWorks, 2019) e impresa gracias a una impresora 3D en

la Universidad Loyola Andalucía. Los planos de dicha carcasa se encuentran en el Anexo V.

En definitiva, se ha conseguido crear una aplicación que funcione, y que cumpla los requisitos planteados inicialmente, es decir, operar desconectada de la red eléctrica y de cualquier red que suponga un gasto periódico como WiFi o 4G; cubrir grandes distancias, por el hecho de que va a ser utilizada en unas instalaciones agrícolas amplias y tener un coste relativamente reducido, teniendo en cuenta los precios existentes en el mercado, ver apartado 2.1. Además, se recoge en este documento gran cantidad de información, así como comparaciones y especificaciones técnicas de dispositivos y tecnologías de comunicaciones en el Internet of Things.

7.2. Futuras líneas de investigación

Tal y como se plantea en los objetivos iniciales, este proyecto pretende servir como base o apoyo para otros proyectos de investigación en el ámbito del Internet de las Cosas, ya sea enfocado al sector agrícola o a otro sector.

En esta sección se exponen algunas posibles líneas de investigación futuras en continuación a este proyecto.

- En referencia a la seguridad del hardware, la electrónica de la placa intermedia podría ser mejorada para aumentar la protección de los circuitos.

- Por otro lado, siguiendo la línea de protección de los componentes, en el caso que aquí se presenta, el microcontrolador espera un tiempo de X segundos predefinido a que se apague de forma segura la Raspberry Pi. Este tiempo es el tiempo calculado que suele tardar la misma en apagarse, sin embargo, para evitar que en algún caso se le corte la alimentación sin que haya tenido tiempo de ejecutar el apagado se propone otra solución más fiable: poner un pin digital de la Raspberry siempre a 1. De esta forma, en el momento en que se apague será 0, y en cuanto el microcontrolador detecte esto, procedería a cortar la alimentación de la Raspberry Pi.

- En este caso, la idea en principio es realizar una foto diariamente, no obstante, podría darse el caso de que se requiera un mayor número de muestras al día. En esta situación, sería necesario añadir un sensor fotoeléctrico al circuito, de forma que no realice fotos ni cálculos cuando sea de noche. Cuando no haya luz las fotos no serán de buena calidad y, en consecuencia, los cálculos efectuados sobre ellas no serán fiables. Aunque es posible filtrar los datos a posteriori por hora en que han sido enviados, dado que en la base de datos se almacena también la hora a la que éstos llegan, supone un gasto energético innecesario de los dispositivos.

- Instalación de una pequeña placa solar además de la batería. Esto permitiría que el conjunto funcionase mucho más tiempo sin necesidad de recarga de la batería.

- Utilizar el modo *DeepSleep* del microcontrolador, con el que sería posible ahorrar aún más energía.

- Investigar sobre soluciones más compactas o integradas, es decir, si sería posible encontrar o diseñar un microcontrolador que pudiera realizar el proceso sin necesidad de la Raspberry Pi. Para ello debe reunir las características imprescindibles para este proyecto de ambos dispositivos: una alta potencia de cálculo, soportar LoRa y tener un bajo consumo. Por ejemplo, podría estudiarse si la matriz de puertas programables FPGA (Field-Programmable Gate Array) podría realizar todo el proceso.

- Estudiar otros protocolos de comunicación IoT. En este proyecto se decidió utilizar LoRaWAN para la comunicación, y a raíz de ahí, se eligió el hardware necesario para soportar este protocolo. No obstante, como ya se ha mencionado, existen otras tecnologías y protocolos como SigFox que podrían ser útiles también en la situación que se plantea.

- Desarrollar una aplicación final en la que almacenar y filtrar los datos que llegan a The Things Network.

7.3. Conclusiones personales

En conclusión, se puede decir que el Internet of Things se encuentra en un continuo y acelerado desarrollo, planteando nuevos horizontes de investigación y de mejora en todos los ámbitos de la vida. La investigación de este concepto y de las tecnologías y dispositivos hardware más utilizados han permitido a la alumna conocer las posibilidades que puede ofrecer el IoT en la industria. Asimismo, este estudio le ha ofrecido nuevas perspectivas de cara a un futuro laboral en este sector que plantea grandes posibilidades de mejora en un futuro no muy lejano.

8. Anexos

Anexo I: Presupuesto del Proyecto

Elaborado por: María Andrades Cózar

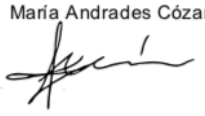
Proyecto: Integración de Raspberry Pi y microcontrolador Pycom para la realización de tareas complejas en un entorno IoT

Duración: 5 meses - 300 horas

Tipo de recurso	Descripción	Tipo de unidad	Unidades	Precio por unidad	Importe
Sofware					
	The Things Network	NA	1	- €	- €
	Python	NA	1	- €	- €
	Atom	NA	1	- €	- €
	OpenCV	NA	1	- €	- €
				Subtotal	**- €**
Hardware de adquisición única					
	Cargador	Euros	1	12,81 €	12,81 €
	Teclado	Euros	1	14,95 €	14,95 €
	Ratón	Euros	1	5,99 €	5,99 €
	Monitor	Euros	1	68,90 €	68,90 €
	Gateway Sentrius RG1xx	Euros	1	217,79 €	217,79 €
	Cable HDMI	Euros	1	4,99 €	4,99 €
	Cable Ethernet	Euros	1	7,45 €	7,45 €
				Subtotal	**332,88 €**
Hardware por nodo					
	Raspberry Pi 3	Euros	1	33,95 €	33,95 €
	Tarjeta micro SD 32 GB	Euros	1	3,00 €	3,00 €
	Pi Camera Module V2 8MP	Euros	1	25,92 €	25,92 €
	Placa de desarrollo LoPy 4 + antena	Euros	1	56,40 €	56,40 €
	Placa de expansión 3.0	Euros	1	16,00 €	16,00 €
	Cable USB-micro USB	Euros	2	2,50 €	5,00 €
	Relé electrónico	Euros	1	4,50 €	4,50 €
	Componentes electrónicos y cables	Euros	1	3,00 €	3,00 €
	Placa PCB	Euros	1	3,50 €	3,50 €
	Power bank	Euros	1	11,00 €	11,00 €
	Hub puertos USB	Euros	1	12,34 €	12,34 €
				Subtotal	**174,61 €**
				Total	507,49 €

El presupuesto total asciende a la cantidad de quinientos siete con cuarenta y nueve euros (507,49 €)

En Sevilla, a 2 de Julio de 2019 Firmado: María Andrades Cózar



Anexo II: Código de Raspberry Pi

```
import time
import picamera
import cv2
import serial
import numpy as np

with picamera.PiCamera() as picam:
    picam.capture('/home/pi/Documents/imagen.jpg')
    picam.close()
imagen = cv2.imread('/home/pi/Documents/imagen.jpg')

hsv = cv2.cvtColor(imagen, cv2.COLOR_BGR2HSV)

verdes_bajos = np.array([49,50,50])
verdes_altos = np.array([80,255,255])

mask = cv2.inRange(hsv, verdes_bajos, verdes_altos)
verde = cv2.countNonZero(mask)
#en hsv el cero es el negro, así que si cuento los que no son
0 estoy contando los verdes
#print(verde)


with serial.Serial('/dev/ttyS0',baudrate=115200, timeout=10,
parity=serial.PARITY_NONE, stopbits=serial.STOPBITS_ONE,
bytesize=serial.EIGHTBITS) as ser:
    ser.write(str(verde))
#print("DATA SENT")
```

Anexo III: Código microcontrolador Pycom

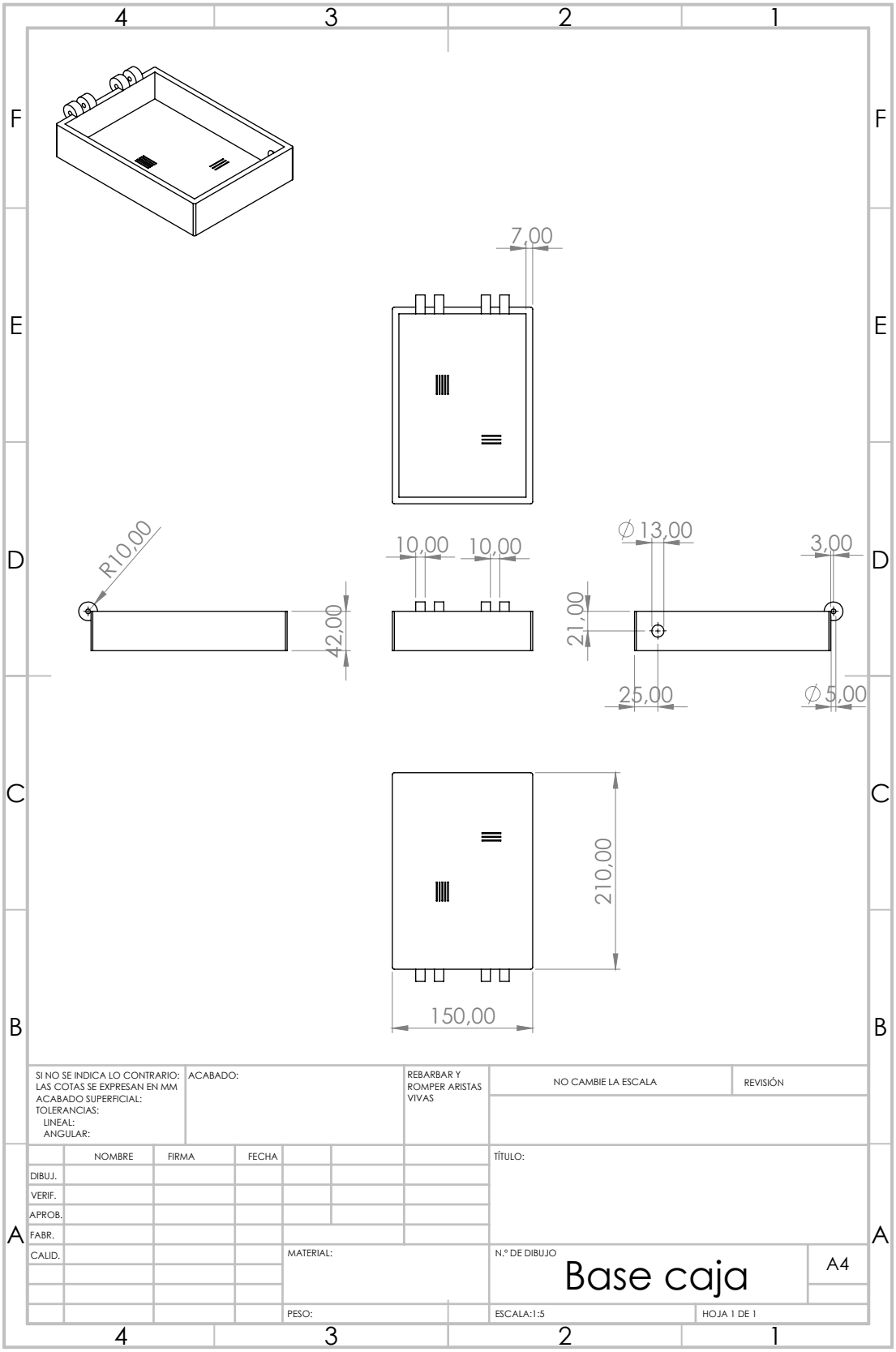
```
import time
import pycom
from machine import UART
from machine import Pin
from network import LoRa
import socket
import ubinascii
import ustruct

p_out = Pin('P10', mode=Pin.OUT)

while True:

    #grabo en la variable starttime la hora de inicio del bucle
    starttime=time.time()

    pycom.heartbeat(False)
    pycom.rgbled(0x000010)
    #enciendo Raspberry Pi
    p_out.hold(False)
    p_out.value(1)
    p_out.hold(True)

    #espero datos de la Raspberry Pi
    uart1 = UART(1, 115200, bits=8, parity=None, stop=1,
pins=(None,'P21'))
    uart1.init(baudrate=115200, bits=8, parity=None, stop=1,
pins=(None, 'P21'))

    flag=1
    while flag==1:
        if uart1.any():
            data = uart1.readall()
            #print(int(data))
            #uart1.deinit()
            flag=0
        else:
            flag=1

    #una vez recibidos, espero a que la RPi se apague
    pycom.rgbled(0xff00)
    time.sleep(21)
    #corto alimentación Raspberry Pi
    p_out.hold(False)
    p_out.value(0)
    p_out.hold(True)
```

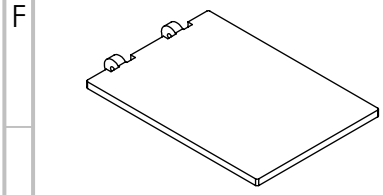
```
    #envío datos a The Things network

    #iniciar LoRa en modo LoRaWAN

    pycom.rgbled(0xFF0000)
    lora=LoRa(mode=LoRa.LORAWAN, region=LoRa.EU868)

    #parámetros para OTAA

    app_eui = ubinascii.unhexlify('70B3D57ED001B1E3')
    app_key                                                    =
ubinascii.unhexlify('AF8845DAB62F0CBA29977E493B436B1C')

    lora.join(activation=LoRa.OTAA,  auth=(app_eui,  app_key),
timeout=0)

    while not lora.has_joined():
        time.sleep(2.5)
        #print('Not yet joined...')
    pycom.rgbled(0xffa500)

    s = socket.socket(socket.AF_LORA, socket.SOCK_RAW)
    s.setsockopt(socket.SOL_LORA, socket.SO_DR, 5)

    s.setblocking(True)
    data=int(data)
    datapack=ustruct.pack('q', data)
    s.send(datapack)

    # enviar datos

    pycom.rgbled(0x00FF00)
    time.sleep(1)
    uart1.deinit()
    pycom.heartbeat(False)
    #paro un minuto antes de reiniciar el bucle
    time.sleep(60.0 - ((time.time() - starttime) % 60.0))
```

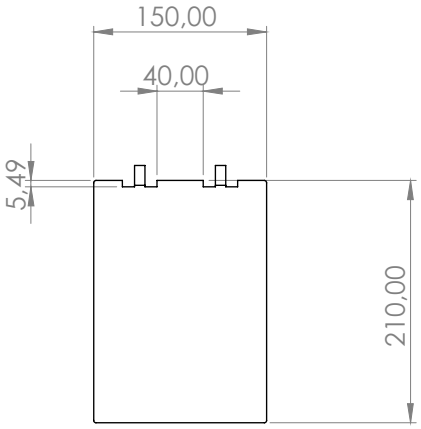
Anexo IV: Código decodificación TTN

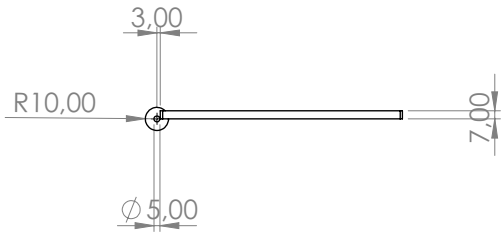
```
function Decoder(bytes, port) {
  // Decode an uplink message from a buffer
  // (array) of bytes to an object of fields.
  var decoded = {};

  // Decode bytes to int
  var entero = bytes[7] << 64 | bytes[6] << 56 | bytes[5] << 48 |
bytes[4] << 40 | bytes[3] << 32 | bytes[2] << 16 | bytes[1] << 8 |
bytes[0];

  // Decode int
  decoded.pixeles = entero;

  return decoded;
}
```

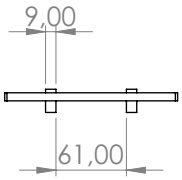
Anexo V: Planos para carcasa en SolidWorks











SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:	ACABADO:		REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN		
	NOMBRE	FIRMA	FECHA	TÍTULO:			
DIBUJ.							
VERIF.							
APROB.							
FABR.							
CALID.				MATERIAL:	N.º DE DIBUJO	A4	
					Tapa		
				PESO:	ESCALA:1:5	HOJA 1 DE 1	

ANEXOS

9. Bibliografía

Organización de las Naciones Unidas para la Alimentación y la Agricultura. (2018). *El futuro de la alimentación y la agricultura: Vías alternativas hacia el 2050. Versión Resumida.* Roma: FAO.

Accent Systems NB IoT. (2019). *Accent Systems NB IoT*. Obtenido de Accent Systems NB IoT: https://accent-systems.com/es/desarrollamos-tus-dispositivos-nb-iot/

Arduino. (2019). *Arduino*. Obtenido de Arduino: https://www.arduino.cc

Atom. (2019). *Atom*. Obtenido de Atom: https://atom.io

Atom Packages. (2019). *Atom Packages*. Obtenido de Atom Packages: pymakr: https://atom.io/packages/pymakr

Barros, J. (10 de Octubre de 2016). *Aprende Fotografía: tamaño de imagen, resolución y tamaño físico*. Obtenido de Jota Barros: https://jotabarros.com/aprende-fotografia-tamano-imagen-resolucion-tamano-fisico/

Barry, R., & Meijers, J. (9 de Enero de 2018). *IoT connectivity comparison (GSM vs LoRa vs Sigfox vs NB-Iot)*. Obtenido de polymorph: https://www.polymorph.co.za/iot-connectivity-comparison-gsm-vs-lora-vs-sigfox-vs-nb-iot/

BeagleBoard.org. (30 de Abril de 2019). *Beagle Board*. Obtenido de BeagleBone Black: https://beagleboard.org/black

Briceño, M. (20 de Febrero de 2018). *Sofia2*. Obtenido de ¿QUÉ ES LA INICIATIVA THE THINGS NETWORK?: https://about.sofia2.com/2018/02/20/que-es-la-iniciativa-the-things-network/

Core Electronics. (25 de Enery de 2019). *Core Electronics*. Obtenido de Encoding and Decoding Payloads on The Things Network: https://core-electronics.com.au/tutorials/encoding-and-decoding-payloads-on-the-things-network.html

Deloitte Insight. (2017). *Forces of Change: Industry 4.0. A Deloitte series on Industry 4.0.* Matthew Budman, Abrar Khan. Obtenido de Deloitte:

https://www2.deloitte.com/es/es/pages/manufacturing/articles/que-es-la-industria-4.0.html

Developer Qualcomm. (2019). *DragonBoard 410c Development Board*. Obtenido de Qualcomm Developer Network: https://developer.qualcomm.com/hardware/dragonboard-410c

Documentation GPIO. (2019). *Raspberry Pi Org.* Obtenido de GPIO: https://www.raspberrypi.org/documentation/usage/gpio/

EcuRed. (18 de Enero de 2015). *EcuRed*. Obtenido de Modelo HSV: https://www.ecured.cu/Modelo_HSV

Espressif Systems. (2019). *Espressif Systems*. Obtenido de Espressif Systems: https://www.espressif.com/en/products/hardware/esp8266ex/overview

Fracttal. (1 de Julio de 2019). *Fracttal*. Obtenido de Las 9 aplicaciones más importantes del Internet de las Cosas (IoT): https://www.fracttal.com/blog/2018/10/10/9-aplicaciones-importantes-iot

Geerling, J. (5 de Abril de 2018). *Raspberry Pi Dramble*. Obtenido de Power Consumption Benchmarks: http://www.pidramble.com/wiki/benchmarks/power-consumption

Grupo Garatu IT Solutions. (27 de Febrero de 2019). *Tecnología 5G en la Industria 4.0 (IoT)*. Obtenido de Grupo Garatu IT Solutions: https://grupogaratu.com/tecnologia-5g-que-es-como-beneficia-industria-4-0-iot/

I. C., D. A., T. P., H. D., & M. C. (21-25 May 2012). *A brief introduction to OpenCV.* Opatija, Croatia: IEEE.

IBM. (5 de Diciembre de 2018). *Conociendo MQTT*. Obtenido de IBM Developer: https://www.ibm.com/developerworks/ssa/library/iot-mqtt-why-good-for-iot/index.html

IBM Developer. (3 de Enero de 2018). *Choosing the best hardware for your next IoT project*. Obtenido de IBM Developer: https://developer.ibm.com/articles/iot-lp101-best-hardware-devices-iot-project/

JavaScript. (2019). *JavaScript*. Obtenido de JavaScript: https://www.javascript.com

Jedermann, R., Hartgenbush, N., Borysov, Jaeger, Sellwig, & Lang. (2018). Testing Lora for Food Applications - Example application for airflow measurements inside cooled warehouses with apples. *Procedia Manufacturing*, 284-289.

Kit electrónica. (2017). Obtenido de Tutorial circuito integrado L293: https://www.kitelectronica.com/2016/02/tutorial-circuito-integrado-l293.html

Laird. (19 de Abril de 2018). *Sentrius RG1xx Quick Start Guide.*

Laird Smart Technology. (2019). *Laird*. Obtenido de Sentrius RG1xx LoRa-Enabled Gateway + Wi-Fi / Bluetooth / Ethernet: https://www.lairdconnect.com/wireless-modules/lorawan-solutions/sentrius-rg1xx-lora-enabled-gateway-wi-fi-bluetooth-ethernet

Link Labs. (25 de Junio de 2018). *NB-IoT vs. LoRa vs. Sigfox*. Obtenido de Link Labs: https://www.link-labs.com/blog/nb-iot-vs-lora-vs-sigfox

LoRa Alliance. (2019). *About LoRa Alliance*. Obtenido de LoRa Alliance: https://lora-alliance.org/about-lora-alliance

LoRaWAN España. (2019). *LoRaWAN*. Obtenido de LoRaWAN: https://lorawan.es

Maulucioni. (2 de Junio de 2015). *Wikipedia.* Obtenido de Cono del sistema de color HSV: https://commons.wikimedia.org/wiki/File:Cono_de_la_coloración_HSV.png

McCelland, C. (11 de Abril de 2017). *LPWAN - The Benefits of LPWAN Technology vs. Other IoT Connectivity Options*. Obtenido de Leverage: https://www.leverege.com/blogpost/lpwan-benefits-vs-iot-connectivity-options

Mekki, K., Bajic, E., Chaxel, F., & Meyer, F. (2019). A comparative study of LPWAN technologies for large-scale IoT deployment. *ICT Express*, Volumen 5, publicación 1, páginas 1-7.

MicroPython. (2019). *MicroPython*. Obtenido de MicroPython: https://micropython.org

Naciones Unidas. (21 de Junio de 2017). *La población mundial aumentará en 1.000 millones para 2030*. Obtenido de Naciones Unidas: Departamento de Asuntos Económicos y Sociales: https://www.un.org/development/desa/es/news/population/world-population-prospects-2017.html

OpenCV. (2019). *About OpenCV*. Obtenido de OpenCV: https://opencv.org/about/

Pycom . (2019). *Pycom Documentation Site*. Obtenido de Pycom go invent: https://docs.pycom.io

Pycom. (2019). *Pycom Products*. Obtenido de Pycom: https://pycom.io/product/lopy4/

Pycom GitBook. (2019). *Pycom go invent*. Obtenido de MQTT: https://docs.pycom.io/tutorials/all/mqtt.html

Pycom Go Invent. (2018). LoPy4 Datasheet Version 1.0. *LoPy4 Specifications Sheet*. Obtenido de LoPy4 Datesheet Version 1.0.

Python Programming. (2019). *PythonProgramming*. Obtenido de PythonProgramming: https://pythonprogramming.net

Raspberry Pi. (2019). *Raspberry Pi*. Obtenido de Raspberry Pi: https://www.raspberrypi.org

RS Components Spain. (2019). *RS Components Spain*. Obtenido de Relé Finder: https://es.rs-online.com/web/p/products/0351617/

S. F., & X. L. (2014). Wireless Sensor Network System Desing using Raspberry Pi and Arduino for Environmental Monitoring Applications. *Procedia Computer Science*, 103-110.

Salvador, H. S. (2018). *The Things Network*. Obtenido de Lanzamiento de The Things Network: https://www.thethingsnetwork.org/community/san-salvador/post/lanzamiento-the-things-network

SigFox España. (2019). *SigFox España*. Obtenido de SigFox España: https://www.sigfox.es

Simform. (20 de Junio de 2018). *LoRaWAN vs Zigbee: Part 1 – How to compare and select*. Obtenido de Simform: https://www.simform.com/lorawan-vs-zigbee-comparison-part-1/

SolidWorks. (2019). *Dassault Systems*. Obtenido de SolidWorks: https://www.solidworks.com/es

Swagger UI. (s.f.). *The Things Network Data Storage*. Obtenido de The Things Network Data Storage: https://tfg_mariaandrades.data.thethingsnetwork.org/#!/devices/get_api_v2_devices

The Things Industries. (2019). *The Things Industries*. Obtenido de The Things Industries: https://www.thethingsindustries.com

The Things Network Community. (2019). *The Things Network*. Obtenido de The Things Network: https://www.thethingsnetwork.org

Universidad de Alcalá. (11 de Marzo de 2019). *IOT Y 5G: LAS TECNOLOGÍAS QUE REVOLUCIONARÁN 2019*. Obtenido de Master Internet of Things: https://www.master-internet-of-things.com/iot-5g-revolucionaran-2019/

V. A., K. B., & S. R. (2017). Implementation of effective and low-cost building monitoring system (BMS) using Raspberry Pi. *Energy Procedia*, 179-185.

Vermesan, O., & Friess, P. (2014). *Internet of Things – From Research and Innovation to Market Deployment.* River Publishers Series in Communication. Obtenido de IERC: Internet of Things: http://www.internet-of-things-research.eu/about_iot.htm

VGD, J. (1 de Enero de 2018). *The Things Network*. Obtenido de ¿Por qué LoRaWAN?: https://www.thethingsnetwork.org/community/sevilla/post/por-que-lorawan

Zafra, P. G. (2019-2021). INNORIEGA (rIego sosteNible basado en moNitOrización distribuída e intEliGencia Artificial). *PP.AVA.AVA2019.024*. España.

ZigBee Alliance. (2019). *ZigBee Alliance*. Obtenido de ZigBee Alliance: https://www.zigbee.org